F 13

```
UUU         UUU   EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP        SSSSSSSSSSSS  YYY         YYY
UUU         UUU   EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP        SSSSSSSSSSSS  YYY         YYY
UUU         UUU   EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP        SSSSSSSSSSSS  YYY         YYY
UUU         UUU   EEE                    TTT        PPP         PPP  SSS              YYY         YYY
UUU         UUU   EEE                    TTT        PPP         PPP  SSS              YYY         YYY
UUU         UUU   EEE                    TTT        PPP         PPP  SSS              YYY         YYY
UUU         UUU   EEE                    TTT        PPP         PPP  SSS                 YYY   YYY
UUU         UUU   EEE                    TTT        PPP         PPP  SSS                 YYY   YYY
UUU         UUU   EEE                    TTT        PPP         PPP  SSS                 YYY   YYY
UUU         UUU   EEEEEEEEEEEE           TTT        PPPPPPPPPPPP        SSSSSSSSS           YYY
UUU         UUU   EEEEEEEEEEEE           TTT        PPPPPPPPPPPP        SSSSSSSSS           YYY
UUU         UUU   EEEEEEEEEEEE           TTT        PPPPPPPPPPPP        SSSSSSSSS           YYY
UUU         UUU   EEE                    TTT        PPP                          SSS        YYY
UUU         UUU   EEE                    TTT        PPP                          SSS        YYY
UUU         UUU   EEE                    TTT        PPP                          SSS        YYY
UUU         UUU   EEE                    TTT        PPP                          SSS        YYY
UUU         UUU   EEE                    TTT        PPP                          SSS        YYY
UUU         UUU   EEE                    TTT        PPP                          SSS        YYY
UUUUUUUUUUUUUUU   EEEEEEEEEEEEEEE        TTT        PPP              SSSSSSSSSSSS           YYY
UUUUUUUUUUUUUUU   EEEEEEEEEEEEEEE        TTT        PPP              SSSSSSSSSSSS           YYY
UUUUUUUUUUUUUUU   EEEEEEEEEEEEEEE        TTT        PPP              SSSSSSSSSSSS           YYY
```

```
_$


Va
--
00
00
00
00
00
48
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
```

```
**FILE**ID**SATSSS01                              M  2

  SSSSSSSS    AAAAAA    TTTTTTTTTT    SSSSSSSS    SSSSSSSS    SSSSSSSS    000000        11
  SSSSSSSS    AAAAAA    TTTTTTTTTT    SSSSSSSS    SSSSSSSS    SSSSSSSS    000000        11
SS          AA      AA      TT      SS          SS          SS          00      00    1111
SS          AA      AA      TT      SS          SS          SS          00      00    1111
SS          AA      AA      TT      SS          SS          SS          00    0000      11
SS          AA      AA      TT      SS          SS          SS          00    0000      11
  SSSSSS    AA      AA      TT        SSSSSS      SSSSSS      SSSSSS    00  00  00      11
  SSSSSS    AA      AA      TT        SSSSSS      SSSSSS      SSSSSS    00  00  00      11
        SS  AAAAAAAAAA      TT              SS          SS          SS  0000    00      11
        SS  AAAAAAAAAA      TT              SS          SS          SS  0000    00      11
        SS  AA      AA      TT              SS          SS          SS  00      00      11          ....
        SS  AA      AA      TT              SS          SS          SS  00      00      11          ....
SSSSSSSS    AA      AA      TT      SSSSSSSS    SSSSSSSS    SSSSSSSS      000000      111111        ....
SSSSSSSS    AA      AA      TT      SSSSSSSS    SSSSSSSS    SSSSSSSS      000000      111111        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
SAT
V04
```

Table of contents

SATS
V04-

```
0000          .TITLE  SATSSS01 - SATS SYSTEM SERVICE TESTS  (SUCC S.C.)
0000          .IDENT  'V04-000'
0000  ;
0000  ;****************************************************************************
0000  ;*                                                                          *
0000  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000  ;*  ALL RIGHTS RESERVED.                                                    *
0000  ;*                                                                          *
0000  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000  ;*  TRANSFERRED.                                                            *
0000  ;*                                                                          *
0000  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000  ;*  CORPORATION.                                                            *
0000  ;*                                                                          *
0000  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000  ;*                                                                          *
0000  ;*                                                                          *
0000  ;****************************************************************************
0000  ;
0000
0000  ;++
0000  ; FACILITY:      SATS SYSTEM SERVICE TESTS
0000  ;
0000  ; ABSTRACT:      The SATSSS01 module tests the execution of the following
0000  ;                VMS system services:
0000  ;
0000  ;                $ASSIGN
0000  ;                $ALLOC
0000  ;                $CANCEL
0000  ;                $DASSGN
0000  ;                $DALLOC
0000  ;                $INPUT
0000  ;                $GETCHN
0000  ;                $GETDEV
0000  ;                $OUTPUT
0000  ;                $QIO
0000  ;                $QIOW
0000  ;
0000  ;
0000  ; ENVIRONMENT:   User mode image.
0000  ;                Needs CMKRNL privilege and dynamically acquires other
0000  ;                privileges, as needed.
0000  ;
0000  ; AUTHOR: Larry D. Jones,              CREATION DATE: JULY, 1979
0000  ;
0000  ; MODIFIED BY:
0000  ;
0000  ;       V03-004 KDM0002         Kathleen D. Morse       28-Jun-1982
0000  ;               Added $PRDEF and $SSDEF.
0000  ;
```

```
                          0000     ;           V03-003 RNH0002         Richard N. Holstein,    22-Jun-1982
                          0000     ;                   Fix to print correct device and unit number when checking data
                          0000     ;                   buffer for disks (STP36).
                          0000     ;**
                          0000     ;--
```

```
                 0000          .SBTTL  DECLARATIONS
                 0000   ;
                 0000   ; MACRO LIBRARY CALLS
                 0000   ;
                 0000          .LIBRARY /SYS$LIBRARY:STARLET.MLB/
                 0000          $ATRDEF                          ; attribute control block definitions
                 0000          $CCBDEF                          ; channel control block definitions
                 0000          $DCDEF                           ; device characteristics definitions
                 0000          $DEVDEF                          ; device definitions
                 0000          $DIBDEF                          ; device information block definitions
                 0000          $DVIDEF                          ; $GETDVI definitions
                 0000          $FIBDEF                          ; file information block definitions
                 0000          $PHDDEF                          ; process header offset definitions
                 0000          $PRDEF                           ; processor register definitions
                 0000          $PRVDEF                          ; privilege definitions
                 0000          $PSLDEF                          ; PSL definitions
                 0000          $SHR_MESSAGES UETP,116,<<TEXT,INFO>> ; UETP$_TEXT definition
                 0000          $SFDEF                           ; stack frame definitions
                 0000          $SSDEF                           ; system status code definitions
                 0000          $STSDEF                          ; STS definitions
                 0000          $UETPDEF                         ; UETP message definitions
                 0000   ;
                 0000   ; Equated symbols
                 0000   ;
00000000         0000   WARNING          = 0                    ; warning severity value for msgs
00000001         0000   SUCCESS          = 1                    ; success      ''     ''   ''   ''
00000002         0000   ERROR            = 2                    ; error        ''     ''   ''   ''
00000003         0000   INFO             = 3                    ; information  ''     ''   ''   ''
00000004         0000   SEVERE           = 4                    ; fatal        ''     ''   ''   ''
                 0000   ;
00040004         0000   MFD_FILE_ID      = <4@16>+4             ; MFD ID
                 0000   ;
                 0000   ; MACROS
                 0000   ;
```

```
                              00000000     99         .PSECT  RODATA,RD,NOWRT,NOEXE,LONG
                                  0000    100 ;
                                  0000    101 TEST_MOD_NAME:
         31 30 53 53 53 54 41 53 00' 0000    102         .ASCIC  /SATSSS01/                ; needed for SATSMS message
                                  08 0000
                                  0009    103 TEST_MOD_NAME_D:
53 53 53 54 41 53 00000011'010E0000' 0009    104         .ASCID  /SATSSS01/                ; module name
                               31 30 0017
                                  0019    105 TEST_MOD_BEGIN:                           ; start end and fail messages
                  6E 75 67 65 62 00' 0019    106         .ASCIC  /begun/
                                  05 0019
                                  001F    107 TEST_MOD_SUCC:
   6C 75 66 73 73 65 63 63 75 73 00' 001F    108         .ASCIC  /successful/
                                  0A 001F
                                  002A    109 TEST_MOD_FAIL:
               64 65 6C 69 61 66 00' 002A    110         .ASCIC  /failed/
                                  06 002A
                                  0031    111 ASSIGN:                                   ; system service names
               4E 47 49 53 53 41 00' 0031    112         .ASCIC  /ASSIGN/
                                  06 0031
                                  0038    113 ALLOC:
                  43 4F 4C 4C 41 00' 0038    114         .ASCIC  /ALLOC/
                                  05 0038
                                  003E    115 CANCEL:
               4C 45 43 4E 41 43 00' 003E    116         .ASCIC  /CANCEL/
                                  06 003E
                                  0045    117 DASSGN:
               4E 47 53 53 41 44 00' 0045    118         .ASCIC  /DASSGN/
                                  06 0045
                                  004C    119 DALLOC:
               43 4F 4C 4C 41 44 00' 004C    120         .ASCIC  /DALLOC/
                                  06 004C
                                  0053    121 INPUT:
                  54 55 50 4E 49 00' 0053    122         .ASCIC  /INPUT/
                                  05 0053
                                  0059    123 GETCHN:
               4E 48 43 54 45 47 00' 0059    124         .ASCIC  /GETCHN/
                                  06 0059
                                  0060    125 GETDEV:
               56 45 44 54 45 47 00' 0060    126         .ASCIC  /GETDEV/
                                  06 0060
                                  0067    127 OUTPUT:
               54 55 50 54 55 4F 00' 0067    128         .ASCIC  /OUTPUT/
                                  06 0067
                                  006E    129 QIO:
                        4F 49 51 00' 006E    130         .ASCIC  /QIO/
                                  03 006E
                                  0072    131 QIOW:
                     57 4F 49 51 00' 0072    132         .ASCIC  /QIOW/
                                  04 0072
                                  0077    133 DCLCMH:
               48 4D 43 4C 43 44 00' 0077    134         .ASCIC  /DCLCMH/
                                  06 0077
                                  007E    135 RENAST:
54 53 54 4F 49 51 00000086'010E0000' 007E    136         .ASCID  /QIOTST.DAT;1     /      ; returned name string
   20 20 20 20 20 31 3B 54 41 44 2E 008C
                                  0097    137 DISK:
```

```
49 44 24 53 59 53 0000009F'010E0000'  0097     138             .ASCID  /SYS$DISK/                      ; qio device name
                               4B 53  00A5
                                      00A7     139 CS1:                                            ; failure messages
21 20 74 73 65 54 000000AF'010E0000'  00A7     140             .ASCID  \Test !AC service name !AC step !UL failed.\
6E 20 65 63 69 76 72 65 73 20 43 41   00B5
70 65 74 73 20 43 41 21 20 65 6D 61   00C1
2E 64 65 6C 69 61 66 20 4C 55 21 20   00CD
                                      00D9     141 CS2:
74 63 65 70 78 45 000000E1'010E0000'  00D9     142             .ASCID  \Expected !AS = !XL received !AS = !XL\
4C 58 21 20 3D 20 53 41 21 20 64 65   00E7
41 21 20 64 65 76 69 65 63 65 72 20   00F3
                  4C 58 21 20 3D 20 53  00FF
                                      0106     143 CS3:
74 63 65 70 78 45 0000010E'010E0000'  0106     144             .ASCID  \Expected !AS!UB = !XL received !AS!UB = !XL\
20 3D 20 42 55 21 53 41 21 20 64 65   0114
64 65 76 69 65 63 65 72 20 4C 58 21   0120
58 21 20 3D 20 42 55 21 53 41 21 20   012C
                                  4C  0138
                                      0139     145 CS4:
72 69 75 71 65 52 00000141'010E0000'  0139     146             .ASCID  \Required channel not received.\
6E 20 6C 65 6E 6E 61 68 63 20 64 65   0147
2E 64 65 76 69 65 63 65 72 20 74 6F   0153
                                      015F     147 CS5:
77 20 65 64 6F 4D 00000167'010E0000'  015F     148             .ASCID  \Mode was !AS.\
                  2E 53 41 21 20 73 61  016D
                                      0174     149 EXP:
73 75 74 61 74 73 0000017C'010E0000'  0174     150             .ASCID  \status\
                                      0182     151 IOEXP:
61 74 73 20 4F 49 0000018A'010E0000'  0182     152             .ASCID  \IO status\
                            73 75 74  0190
                                      0193     153 ASTEXP:
61 70 20 54 53 41 0000019B'010E0000'  0193     154             .ASCID  \AST param.\
                         2E 6D 61 72  01A1
                                      01A5     155 DISALL:
61 20 6B 73 69 64 000001AD'010E0000'  01A5     156             .ASCID  \disk alloc.\
                         2E 63 6F 6C 6C  01B3
                                      01B8     157 IOCC:
63 20 66 6F 20 23 000001C0'010E0000'  01B8     158             .ASCID  \# of chan's\
                      73 27 6E 61 68  01C6
                                      01CB     159 FILNOTMOD:
63 20 65 6C 69 46 000001D3'010E0000'  01CB     160             .ASCID  \File characteristics not properly modified!\
69 74 73 69 72 65 74 63 61 72 61 68   01D9
65 70 6F 72 70 20 74 6F 6E 20 73 63   01E5
64 65 69 66 69 64 6F 6D 20 79 6C 72   01F1
                                  21  01FD
                                      01FE     161 UM:                                             ; mode messages
      72 65 73 75 00000206'010E0000'  01FE     162             .ASCID  \user\
                                      020A     163 SM:
   72 65 70 75 73 00000212'010E0000'  020A     164             .ASCID  \super\
                                      0217     165 EM:
74 75 63 65 78 65 0000021F'010E0000'  0217     166             .ASCID  \executive\
                            65 76 69  0225
                                      0228     167 KM:
6C 65 6E 72 65 6B 00000230'010E0000'  0228     168             .ASCID  \kernel\
                                      0236     169 MBA:                                            ; mailbox name
         41 42 4D 0000023E'010E0000'  0236     170             .ASCID  \MBA\
                                      0241     171 EFCNAM:                                         ; common EFC name
```

```
45 24 50 54 45 55 00000249'010E0000' 0241           .ASCID  \UETP$EF\
                                  46 024F
                                     0250   TEST_DATA:                                  ; QIO test data
                            00000000 0250           A=0
                                     0250           .REPT 132
                                     0250           .BYTE A
                                     0250           A=A+1
                                  00 0250           .ENDR
                                     02D4   ARGLST:
                            00000001 02D4           .LONG   1                           ; super mode setup arg list
                            0000118B' 02D8          .ADDRESS SUPER_MODE
                                     02DC   MSGVEC:
                            00000003 02DC           .LONG   3                           ; PUTMSG message vector
                            00741133 02E0           .LONG   UETP$_TEXT
                            00000001 02E4           .LONG   1
                            000002FF' 02E8          .ADDRESS MESSAGEL
```

```
                               02EC  ;
                               02EC          .SBTTL  R/W PSECT
                           00000000          .PSECT  RWDATA,RD,WRT,NOEXE,LONG
                               0000  ;
                               0000  TPID:
                    00000000   0000          .LONG   0                        ; PID for this process
                               0004  CURRENT_TC:
                    00000000   0004          .LONG   0                        ; ptr to current test case
                               0008          .ALIGN LONG                      ; put it on a long word boundry
                               0008  REG_SAVE_AREA:
                    00000044   0008          .BLKL   15                       ; register save area
                               0044  MOD_MSG_CODE:
                    007480D9   0044          .LONG   UETP$_SATSMS             ; test module message code for putmsg
                               0048  TMN_ADDR:
                    00000000'  0048          .ADDRESS TEST_MOD_NAME
                               004C  TMD_ADDR:
                    00000019'  004C          .ADDRESS TEST_MOD_BEGIN
                               0050  PRVPRT:
                          00   0050          .BYTE   0                        ; protection return byte for SETPRT
                               0051  PRIVMASK:
           00000000 00000000   0051          .QUAD   0                        ; priv. mask
                               0059  CHM_CONT:
                    00000000   0059          .LONG   0                        ; change mode continue address
                               005D  RETADR:
                    00000065   005D          .BLKL   2                        ; returned address's from SETPRT
                               0065  STATUS:
                    00000000   0065          .LONG   0
                               0069  STAT:
                    00000071   0069          .BLKL   2                        ; IO status blk's
                               0071  STAT1:
                    00000079   0071          .BLKL   2
                               0079  ASGN:
                               0079          $ASSIGN MBNAM,CHAN2,PSL$C_USER,0 ; ASSIGN parameter list
                               008D  ALLO:
                               008D          $ALLOC MBNAM,ML,GETBUF,PSL$C_USER ; ALLOC parameter list
                               00A5  CANC:
                               00A5          $CANCEL MBCHAN                   ; CANCEL parameter list
                               00AD  DASS:
                               00AD          $DASSGN 0                        ; DASSGN parameter list
                               00B5  DALL:
                               00B5          $DALLOC MBNAM,PSL$C_USER         ; DALLOC parameter list
                               00C1  GETC:
                               00C1          $GETCHN 0,PL,PB,SL,SB            ; GETCHN parameter list
                               00D9  GETD:
                               00D9          $GETDEV MBNAM,PL,PB,SL,SB        ; GETDEV parameter list
                               00F1  QIOP:
                               00F1          $QIO    31,CHAN1,IO$_READVBLK,STAT1,0,0,GETBUF+8,80,0,0,0,0 ; QIO parameter'
                               0125  QIOWP:
                               0125          $QIOW   31,MBCHAN,IO$_READVBLK,STAT1,0,0,GETBUF+8,80,0,0,0,0 ; QIOW param's
                               0159  MODE:
                    00000000   0159          .LONG   0                        ; current mode string pointer
                               015D  REG:
74 73 69 67 65 72 00000165'010E0000'  015D          .ASCID  \register R\
                 52 20 72 65   016B
                               016F  REGNUM:
                    00000000   016F          .LONG   0                        ; register number
                               0173  MSGL:
```

```
                       00000050  0173          .LONG   80                        ; buffer desc.
                       0000017B' 0177          .ADDRESS BUF
                                 017B  BUF:
                       000001CB  017B          .BLKB   80
                                 01CB  ML:
                       00000000  01CB          .LONG   0                         ; desc. for BUF_CHECK routine
                       000001DB' 01CF          .ADDRESS GETBUF+8
                                 01D3  GETBUF:
                       00000084  01D3          .LONG   132                       ; same as above
                       000001DB' 01D7          .ADDRESS .+4
                       0000025F  01DB          .BLKB   132
                                 025F  CTRSTR:
                       00000084  025F          .LONG   132                       ; same as above
                       00000267' 0263          .ADDRESS .+4
                       000002EB  0267          .BLKB   132
                                 02EB  ARGLST1:                                  ; argument list for BUF_CHECK
                       00000236' 02EB          .ADDRESS MBA
                       000002FF  02EF          .BLKL   4
                                 02FF  MESSAGEL:
                       00000000  02FF          .LONG   0                         ; message desc.
                       0000017B' 0303          .ADDRESS BUF
                                 0307  SERV_NAME:
                       00000000  0307          .LONG   0                         ; service name pointer
                                 030B  PRVHND1:
                       00000000  030B          .LONG   0                         ; previous handler address 1
                                 030F  MBNAM:
4D 24 50 54 45 55 00000317'010E0000' 030F      .ASCID  /UETP$MB/                 ; logical name for mailbox
                              42 031D
                                 031E  MBCHAN:
                            0000 031E          .WORD   0                         ; mailbox channel number
                                 0320  CHAN1:
                            0000 0320          .WORD   0                         ; utility channel numbers
                                 0322  CHAN2:
                            0000 0322          .WORD   0
                                 0324  CHAN_SAVE:
                            0000 0324          .WORD   0                         ; channel count save location
                                 0326  MSGVEC1:                                  ; PUTMSG message vector
                       00000003  0326          .LONG   3
                       00741133  032A          .LONG   UETP$_TEXT
                       00000001  032E          .LONG   1
                       00000000  0332          .LONG   0
                                 0336  MB_DEV_CHAR:
                       0C150001  0336          .LONG   DEV$M_SHR!DEV$M_REC!DEV$M_AVL!DEV$M_IDV!DEV$M_ODV!DEV$M_MBX ;device
                              A0 033A          .BYTE   DC$_MAILBOX               ; device class
                              01 033B          .BYTE   DT$_MBX                   ; device type
                            0100 033C          .WORD   256                       ; buffer size
                       00000000  033E          .LONG   0                         ; device dependent info.
                      0024 0000  0342          .WORD   0,36                      ; unit # & device name offset
                       00000000  0346          .LONG   0                         ; PID
                       00010007  034A          .LONG   ^X10007                   ; owner UIC
                       00000000  034E          .LONG   0                         ; volume protection & error cnt
                       00000000  0352          .LONG   0                         ; operation count
                       00000000  0356          .LONG   0                         ; volume name offset & record size
                    41 42 4D 00' 035A          .ASCIC  /MBA/                     ; device name
                              03 035A
                       00000028  035E          MB_CHAR_SIZE=.-MB_DEV_CHAR
                                 035E  PL:
```

```
                                                  J 3
SATSSS01                        - SATS SYSTEM SERVICE TESTS  (SUCC S.C.) 16-SEP-1984 00:44:47  VAX/VMS Macro V04-00       Page   9
V04-000                          R/W PSECT                                5-SEP-1984 04:29:37  [UETPSY.SRC]SATSSS01.MAR;1      (1)

                          00000000  035E            .LONG   0
                                    0362  SL:
                          00000000  0362            .LONG   0
                                    0366  PB:
                          00000074  0366            .LONG   DIB$K_LENGTH
                          0000036E' 036A            .ADDRESS .+4
                          000003E2  036E            .BLKB   DIB$K_LENGTH
                                    03E2  SB:
                          00000074  03E2            .LONG   DIB$K_LENGTH
                          000003EA' 03E6            .ADDRESS .+4
                          0000045E  03EA            .BLKB   DIB$K_LENGTH
                                    045E  FIBDES:
                          00000029' 045E            .LONG   FIBSIZE                    ; file information block desc.
                          00000466' 0462            .ADDRESS FIB
                                    0466  FIB:
                          00000000  0466            .LONG   0               ; ACCTL
                          00000470  046A            .BLKW   3               ; FID
                          00040004  0470            .LONG   MFD_FILE_ID     ; DID
                          0000048F  0474            .BLKB   27                         ; leave room for add in fields
                          00000029  048F            FIBSIZE=.-FIB                      ; set FIB size
                                    048F  ATR:
                         0010 0056  048F            .WORD   ATR$S_ASCNAME,ATR$C_ASCNAME ; attributes control block
                          000004E4' 0493            .ADDRESS TOPSYS_DIR
                          00000000  0497            .LONG   0
                                    049B  FILENAME:
54 53 54 4F 49 51 000004A3'010E0000' 049B           .ASCID  /QIOTST.DAT;1/             ; qio test file name
            31 3B 54 41 44 2E       04A9
                                    04AF  SYSTEST_DIR:
53 45 54 53 59 53 000004B7'010E0000' 04AF           .ASCID  /SYSTEST.DIR;1/            ; SYSTEST directory name
         31 3B 52 49 44 2E 54       04BD
                                    04C4  DOT_DIR_SEMI:                                ; Concatenates with TOPSYS_DIR
31 3B 52 49 44 2E 000004CC'010E0000' 04C4           .ASCID  /.DIR;1/
                          00000006  04D2  DOT_DIR_SEMI_LENGTH = .-DOT_DIR_SEMI-8       ; Length of ASCII string
                                    04D2  TOPSYS:                                      ; Logical name of any top level...
4F 54 24 53 59 53 000004DA'010E0000' 04D2           .ASCID  /SYS$TOPSYS/               ; ...system directory name
               53 59 53 50          04E0
                                    04E4  TOPSYS_DIR:                                  ; Receives file name of top level...
                          0000000F  04E4            .LONG   9+DOT_DIR_SEMI_LENGTH      ; ...system directory...
                          000004EC' 04E8            .ADDRESS .+4                       ; ...and gets converted to...
                          000004FB  04EC            .BLKB   9+DOT_DIR_SEMI_LENGTH      ; ...a file spec for it
```

```
00000000         .PSECT  SATSSS01,RD,WRT,EXE,LONG
    0000         .SBTTL  SATSSS01
    0000   ;++
    0000   ; FUNCTIONAL DESCRIPTION:
    0000   ;
    0000   ;       After performing some initial housekeeping, such as
    0000   ; printing the module begin message and acquiring needed privileges,
    0000   ; the system services are tested in each of their normal conditions.
    0000   ; Detected failures are identified and  an error message is printed
    0000   ; on the terminal.  Upon completion of the test a success or fail
    0000   ; message is printed on the terminal.
    0000   ;
    0000   ; CALLING SEQUENCE:
    0000   ;
    0000   ;       $ RUN SATSSS01  ...  (DCL COMMAND)
    0000   ;
    0000   ; INPUT PARAMETERS:
    0000   ;
    0000   ;       none
    0000   ;
    0000   ; IMPLICIT INPUTS:
    0000   ;
    0000   ;       none
    0000   ;
    0000   ; OUTPUT PARAMETERS:
    0000   ;
    0000   ;       none
    0000   ;
    0000   ; IMPLICIT OUTPUTS:
    0000   ;
    0000   ;       Messages to SYS$OUTPUT are the only output from SATSSS01.
    0000   ;       They are of the form:
    0000   ;
    0000   ;               %UETP-S-SATSMS, TEST MODULE SATSSS01 BEGUN ... (BEGIN MSG)
    0000   ;               %UETP-S-SATSMS, TEST MODULE SATSSS01 SUCCESSFUL ... (END MSG)
    0000   ;               %UETP-E-SATSMS, TEST MODULE SATSSS01 FAILED ... (END MSG)
    0000   ;               %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
    0000   ;
    0000   ; COMPLETION CODES:
    0000   ;
    0000   ;       The SATSSS01 routine terminates with a $EXIT to the
    0000   ;       operating system with a status code defined by UETP$_SATSMS.
    0000   ;
    0000   ; SIDE EFFECTS:
    0000   ;
    0000   ;       none
    0000   ;
    0000   ;--
    0000
    0000           TEST_START SATSSS01                     ; let the test begin
```

```
                              0000
                        0000  0000                      .ENTRY SATSSS01,0
           0004'CF        D4  0002                      CLRL    W^CURRENT_TC
                00        DD  0006                      PUSHL   #0
           0000'CF        DF  0008                      PUSHAL  W^TPID
   00000000'GF    02      FB  000C                      CALLS   #2,G^SYS$WAKE
   00000000'GF    00      FB  0013                      CALLS   #0,G^SYS$HIBER
           0009'CF        7F  001A                      PUSHAQ  W^TEST_MOD_NAME_D
   00000000'GF    01      FB  001E                      CALLS   #1,G^SYS$SETPRN
              1CC1        30  0025                      BSBW    W^MOD_MSG_PRINT
      004C'CF   001F'CF   DE  0028                      MOVAL   W^TEST_MOD_SUCC,W^TMD_ADDR
0044'CF   03    00    01  F0  002F                      INSV    #SUCCESS,#0,#3,W^MOD_MSG_CODE
                00        DD  0036                      PUSHL   #0
          1385'CF   01    FB  0038                      CALLS #1,W^REG_SAVE
                              003D       STP0:
                              003D            $CMKRNL_S W^SETUP_SUPER,W^ARGLST ; declare CHMS handler
                5E    10'  C0  004C            ADDL2   S^#EXE$C_CMSTKSZ+16,SP   ; adjust the user stack pointer
                5D    5E   D0  004F            MOVL    SP,FP                    ; fix the frame pointer
          1AF2'CF   00    FB  0052            CALLS   #0,W^ERLBUF_DUMP         ; dump any errors that occured at kernal mod
                              0057            .SBTTL  ASSIGN AND DASSGN TESTS
                              0057       ;+
                              0057       ;
                              0057       ; $ASSIGN and $DASSGN tests
                              0057       ;
                              0057       ; ** NOTE **
                              0057       ;
                              0057       ; Because the only device that is reasonable to use for the ASSIGN/DASSGN
                              0057       ; tests is a mailbox, the MBXNAM parameter is not tested by this program.
                              0057       ; The only devices using this parameter are lineprinters, networks,
                              0057       ; and terminals and none of these things can be guaranteed available.
                              0057       ;
                              0057       ; test user mode
                              0057       ;
                              0057       ;-
      0307'CF    0031'CF   DE  0057            MOVAL   W^ASSIGN,W^SERV_NAME             ; set service name
      0159'CF    01FE'CF   DE  005E            MOVAL   W^UM,W^MODE                      ; set mode
                              0065            $ASSIGN_S CHAN = W^MBCHAN,-
                              0065                      DEVNAM = W^MBNAM                ; see if perm MBX left over
      00000908 8F    50    D1  0076            CMPL    R0,#SS$_NOSUCHDEV                ; is it here
                     18    13  007D            BEQL    10$                              ; br if not
                              007F            $DELMBX_S CHAN = W^MBCHAN                ; else get rid of it
                              008B            $DASSGN_S CHAN = W^MBCHAN                ; drop the channel
                              0097       10$:
          133C'CF   00    FB  0097            CALLS   #0,W^COUNT_CHAN                  ; get enviromental channel count
      0324'CF    1338'CF   D0  009C            MOVL    W^TOTAL_CHAN,W^CHAN_SAVE         ; save the enviromental chan count
                03        DD  00A3            PUSHL   #PSL$C_USER                      ; push the access mode
          1BEF'CF   01    FB  00A5            CALLS   #1,W^ASSDAS_CHK                  ; do the assign/deassign tests
          1AF2'CF   00    FB  00AA            CALLS   #0,W^ERLBUF_DUMP                 ; dump any errors
                              00AF       ;+
                              00AF       ;
                              00AF       ; test super mode
                              00AF       ;
                              00AF       ;-
                              00AF            NEXT_TEST
                              00AF
                              00AF       STP1:
           0004'CF   01   D0  00AF                      MOVL    #1,W^CURRENT_TC
```

```
                00   DD   00B4                        PUSHL   #0
        1385'CF 01   FB   00B6                        CALLS   #1,W^REG_SAVE
0307'CF 0031'CF      DE   00BB             MOVAL   W^ASSIGN,W^SERV_NAME              ; set service name
0159'CF 020A'CF      DE   00C2             MOVAL   W^SM,W^MODE                       ; set the mode
                01   BE   00C9             CHMS    #1                                ; do the super tests
        1AF2'CF 00   FB   00CB             CALLS   #0,W^ERLBUF_DUMP                  ; dump any errors
                          00D0     ;+
                          00D0     ;
                          00D0     ; test exec mode
                          00D0     ;
                          00D0     ;-
                          00D0             NEXT_TEST
                          00D0
                          00D0     STP2:
        0004'CF 02   D0   00D0                        MOVL    #2,W^CURRENT_TC
                00   DD   00D5                        PUSHL   #0
        1385'CF 01   FB   00D7                        CALLS   #1,W^REG_SAVE
0159'CF 0217'CF      DE   00DC             MOVAL   W^EM,W^MODE                       ; set the mode
0307'CF 0031'CF      DE   00E3             MOVAL   W^ASSIGN,W^SERV_NAME              ; set service name
                          00EA             $CMEXEC_S B^10$                           ; get thee to exec mode
           000A      31   00F6             BRW     20$
                          00F9     10$:
                0000      00F9             .WORD   0
                01   DD   00FB             PUSHL   #PSL$C_EXEC                       ; push the access mode
        1BEF'CF 01   FB   00FD             CALLS   #1,W^ASSDAS_CHK                   ; do the assign/dassgn tests
                04        0102             RET                                       ; return to user
                          0103     20$:
        1AF2'CF 00   FB   0103             CALLS   #0,W^ERLBUF_DUMP                  ; dump any errors
                          0108     ;+
                          0108     ;
                          0108     ; test kernel mode
                          0108     ;
                          0108     ;-
                          0108             NEXT_TEST
                          0108
                          0108     STP3:
        0004'CF 03   D0   0108                        MOVL    #3,W^CURRENT_TC
                00   DD   010D                        PUSHL   #0
        1385'CF 01   FB   010F                        CALLS   #1,W^REG_SAVE
0307'CF 0031'CF      DE   0114             MOVAL   W^ASSIGN,W^SERV_NAME              ; set service name
0159'CF 0228'CF      DE   011B             MOVAL   W^KM,W^MODE                       ; set the mode
0307'CF 0031'CF      DE   0122             MOVAL   W^ASSIGN,W^SERV_NAME              ; set service name
                          0129             $CMKRNL_S B^10$
           000A      31   0135             BRW     20$                               ; skip the routine
                          0138     10$:
                0000      0138             .WORD   0
                00   DD   013A             PUSHL   #PSL$C_KERNEL                     ; push the access mode
        1BEF'CF 01   FB   013C             CALLS   #1,W^ASSDAS_CHK                   ; do the assign/dassgn tests
                          0141
                04        0141             RET                                       ; return to user mode
                          0142     20$:
        1AF2'CF 00   FB   0142             CALLS #0,W^ERLBUF_DUMP                    ; report any errors
0159'CF 01FE'CF      DE   0147             MOVAL   W^UM,W^MODE                       ; reset the mode
```

```
                             014E               .SBTTL  ALLOC  AND DALLOC TESTS
                             014E         ;+
                             014E         ;
                             014E         ; $ALLOC and $DALLOC tests
                             014E         ;
                             014E         ; test user mode
                             014E         ;
                             014E         ;-
                             014E               NEXT_TEST
                             014E
                             014E         STP4:
       0004'CF    04    D0   014E                      MOVL    #4,W^CURRENT_TC
                  00    DD   0153                      PUSHL   #0
       1385'CF    01    FB   0155                      CALLS   #1,W^REG_SAVE
0307'CF   0038'CF       DE   015A               MOVAL   W^ALLOC,W^SERV_NAME             ; set service name
0159'CF   01FE'CF       DE   0161               MOVAL   W^UM,W^MODE                     ; set the mode
                             0168               $CREMBX_S CHAN  =W^MBCHAN,-
                             0168                         LOGNAM=W^MBNAM,-
                             0168                         PRMFLG=#1                     ; create an allocatable device
                             017F               $DASSGN_S CHAN=W^MBCHAN                 ; make it allocatable
                  03    DD   018B               PUSHL   #PSL$C_USER                     ; push the mode
       1B5C'CF    01    FB   018D               CALLS   #1,W^ALLDAL_CHK                 ; check the services
       1AF2'CF    00    FB   0192               CALLS   #0,W^ERLBUF_DUMP                ; dump any errors
                             0197         ;+
                             0197         ;
                             0197         ; test super mode
                             0197         ;
                             0197         ;-
                             0197               NEXT_TEST
                             0197
                             0197         STP5:
       0004'CF    05    D0   0197                      MOVL    #5,W^CURRENT_TC
                  00    DD   019C                      PUSHL   #0
       1385'CF    01    FB   019E                      CALLS   #1,W^REG_SAVE
0307'CF   0038'CF       DE   01A3               MOVAL   W^ALLOC,W^SERV_NAME             ; set service name
0159'CF   020A'CF       DE   01AA               MOVAL   W^SM,W^MODE                     ; set the mode
                  03    BE   01B1               CHMS    #3                              ; do the super mode tests
                             01B3         ;+
                             01B3         ;
                             01B3         ; test exec mode
                             01B3         ;
                             01B3         ;-
                             01B3               NEXT_TEST
                             01B3
                             01B3         STP6:
       0004'CF    06    D0   01B3                      MOVL    #6,W^CURRENT_TC
                  00    DD   01B8                      PUSHL   #0
       1385'CF    01    FB   01BA                      CALLS   #1,W^REG_SAVE
0307'CF   0038'CF       DE   01BF               MOVAL   W^ALLOC,W^SERV_NAME             ; set service name
0159'CF   0217'CF       DE   01C6               MOVAL   W^EM,W^MODE                     ; set the mode
                             01CD               $CMEXEC_S B^10$                         ; get to exec mode
                  0A    11   01D9               BRB     20$                             ; skip the routine
                             01DB         10$:
                0000         01DB               .WORD   0
                  01    DD   01DD               PUSHL   #PSL$C_EXEC                     ; push the mode
       1B5C'CF    01    FB   01DF               CALLS   #1,W^ALLDAL_CHK                 ; do the tests
                  04         01E4               RET                                     ; return to user mode
```

```
                             01E5     20$:
                             01E5     ;+
                             01E5     ;
                             01E5     ; test kernel mode
                             01E5     ;
                             01E5     ;-
                             01E5             NEXT_TEST
                             01E5
                             01E5     STP7:
           0004'CF    07   D0   01E5                    MOVL    #7,W^CURRENT_TC
                      00   DD   01EA                    PUSHL   #0
           1385'CF    01   FB   01EC                    CALLS   #1,W^REG_SAVE
      0307'CF  0038'CF     DE   01F1            MOVAL   W^ALLOC,W^SERV_NAME             ; set the service name
      0159'CF  0228'CF     DE   01F8            MOVAL   W^KM,W^MODE                     ; set the mode
                                01FF            $CMKRNL_S B^10$                         ; get into kernel mode
                      0A   11   020B            BRB     20$                             ; skip the routine
                                020D  10$:
                         0000   020D            .WORD   0
                      00   DD   020F            PUSHL   #PSL$C_KERNEL                   ; push the mode
           1B5C'CF    01   FB   0211            CALLS   #1,W^ALLDAL_CHK                 ; do the tests
                           04   0216            RET                                     ; return
                                0217  20$:
                                0217            $ASSIGN_S DEVNAM=W^MBNAM,-
                                0217                      CHAN  =W^MBCHAN               ; get the device back
                                0228            $DELMBX_S CHAN  =W^MBCHAN               ; and get rid of it!
  0084 8F    00   01D3'CF    00   2C   0234     MOVC5   #0,W^GETBUF,#0,#132,W^GETBUF+8  ; clean up the buffer
                   01DB'CF              023D
```

```
                             0240            .SBTTL  CANCEL TESTS
                             0240      ;+
                             0240      ;
                             0240      ; $CANCEL tests
                             0240      ;
                             0240      ; test EF wait IO cancellation with _S form
                             0240      ;
                             0240      ;-
                             0240            NEXT_TEST
                             0240
                             0240      STP8:
          0004'CF   08   D0  0240                      MOVL    #8,W^CURRENT_TC
                    00   DD  0245                      PUSHL   #0
          1385'CF   01   FB  0247                      CALLS   #1,W^REG_SAVE
  0307'CF   003E'CF      DE  024C            MOVAL   W^CANCEL,W^SERV_NAME            ; set service name
  0159'CF   01FE'CF      DE  0253            MOVAL   W^UM,W^MODE                     ; set the mode
                             025A            $CREMBX_S CHAN = W^CHAN1,-
                             025A                      LOGNAM = W^MBNAM              ; make a MBX
  00F9'CF   0320'CF      3C  0271            MOVZWL  W^CHAN1,W^QIOP+QIO$_CHAN        ; set the channel up
  00A9'CF   0320'CF      3C  0278            MOVZWL  W^CHAN1,W^CANC+CANCEL$_CHAN     ; in QIO and CANCEL
                             027F            $QIO_G  W^QIOP                          ; do a read on the MBX
                             0288            $CANCEL_S CHAN=W^CHAN1                  ; cancel the IO
                             0294            FAIL_CHECK SS$_NORMAL                   ; check for success
                    01   DD  0294                      PUSHL   #SS$_NORMAL
          138F'CF   01   FB  0296                      CALLS   #1,W^REG_CHECK
                             029B            $WAITFR_S EFN=#31                       ; wait for IO completion
          131B'CF   00   FB  02A4            CALLS   #0,W^CAN_CHECK                  ; check IO status block
                             02A9      ;+
                             02A9      ;
                             02A9      ; test EF wait IO cancellation with _S form
                             02A9      ;
                             02A9      ;-
                             02A9            NEXT_TEST
                             02A9
                             02A9      STP9:
          0004'CF   09   D0  02A9                      MOVL    #9,W^CURRENT_TC
                    00   DD  02AE                      PUSHL   #0
          1385'CF   01   FB  02B0                      CALLS   #1,W^REG_SAVE
                             02B5            $QIO_G  W^QIOP                          ; do a read on the MBX
                             02BE            $CANCEL_G W^CANC                        ; try _G
                             02C7            FAIL_CHECK SS$_NORMAL                   ; check for success
                    01   DD  02C7                      PUSHL   #SS$_NORMAL
          138F'CF   01   FB  02C9                      CALLS   #1,W^REG_CHECK
                             02CE            $WAITFR_S EFN=#31                       ; wait for IO completion
          131B'CF   00   FB  02D7            CALLS   #0,W^CAN_CHECK                  ; check the IO status block
                             02DC      ;+
                             02DC      ;
                             02DC      ; test AST wait IO cancellation with _S form
                             02DC      ;
                             02DC      ;-
                             02DC            NEXT_TEST
                             02DC
                             02DC      STP10:
          0004'CF   0A   D0  02DC                      MOVL    #10,W^CURRENT_TC
                    00   DD  02E1                      PUSHL   #0
          1385'CF   01   FB  02E3                      CALLS   #1,W^REG_SAVE
  0105'CF   1309'CF      DE  02E8            MOVAL   W^IONC,W^QIOP+QIO$_ASTADR       ; set AST address
```

```
                                   02EF    570          $QIO_G  W^QIOP                                 ; issue read on the MBX
                                   02F8    571          $CANCEL_S CHAN=W^CHAN1                          ; cancel it
                                   0304    572          FAIL_CHECK SS$_NORMAL                           ; check success
                     01      DD    0304                         PUSHL   #SS$_NORMAL
          138F'CF    01      FB    0306                         CALLS   #1,W^REG_CHECK
                                   030B    573          $HIBER_S                                        ; wait for AST
                                   0312    574  ;+
                                   0312    575  ;
                                   0312    576  ; test AST wait IO cancellation with _G form
                                   0312    577  ;
                                   0312    578  ;-
                                   0312    579          NEXT_TEST
                                   0312
                                   0312         STP11:
          0004'CF    0B      D0    0312                         MOVL    #11,W^CURRENT_TC
                     00      DD    0317                         PUSHL   #0
          1385'CF    01      FB    0319                         CALLS   #1,W^REG_SAVE
                                   031E    580          $QIO_G  W^QIOP                                 ; issue read to the MBX
                                   0327    581          $CANCEL_G W^CANC                                ; cancel it
                                   0330    582          FAIL_CHECK      SS$_NORMAL                      ; check for success
                     01      DD    0330                         PUSHL   #SS$_NORMAL
          138F'CF    01      FB    0332                         CALLS   #1,W^REG_CHECK
                                   0337    583          $HIBER_S                                        ; wait for AST
          0111'CF    01      D0    033E    584          MOVL    #1,W^QIOP+QIO$_P2                       ; reset QIO parameters
 00FD'CF   00000031'8F       D0    0343    585          MOVL    #IO$_READVBLK,@^QIOP+QIO$_FUNC
             0105'CF         D4    034C    586          CLRL    W^QIOP+QIO$_ASTADR
                                   0350    587          $DASSGN_S CHAN = W^CHAN1                        ; drop the MBX
```

```
                                 035C           .SBTTL  GETCHN TESTS
                                 035C    ;+
                                 035C    ;
                                 035C    ; $GETCHN tests
                                 035C    ;
                                 035C    ; test _S form
                                 035C    ;
                                 035C    ;-
                                 035C           NEXT_TEST
                                 035C
                                 035C    STP12:
          0004'CF    0C    D0    035C                   MOVL    #12,W^CURRENT_TC
                     00    DD    0361                   PUSHL   #0
          1385'CF    01    FB    0363                   CALLS   #1,W^REG_SAVE
     0307'CF   0059'CF     DE    0368           MOVAL   W^GETCHN,W^SERV_NAME            ; set service name
     0159'CF   01FE'CF     DE    036F           MOVAL   W^UM,W^MODE                     ; set the mode
               0069'CF     D4    0376           CLRL    W^STAT                          ; set dummy status
               0071'CF     D4    037A           CLRL    W^STAT1                         ; in #1 & #2
                                 037E           $CREMBX_S   CHAN=W^MBCHAN,-
                                 037E                   PRMFLG=#0,-
                                 037E                   LOGNAM=W^MBNAM                  ; make a device to look at
     00F9'CF   031E'CF     3C    0395           MOVZWL  W^MBCHAN,W^QIOP+QIO$_CHAN       ; save the channel number
                                 039C           $GETCHN_S CHAN  =W^MBCHAN,-
                                 039C                   PRILEN=W^PL,-
                                 039C                   PRIBUF=W^PB,-
                                 039C                   SCDLEN=W^SL,-
                                 039C                   SCDBUF=W^SB                     ; try the _S
                                 03B8           FAIL_CHECK SS$_NORMAL                   ; check success
                     01    DD    03B8                   PUSHL   #SS$_NORMAL
          138F'CF    01    FB    03BA                   CALLS   #1,W^REG_CHECK
               037A'CF     B0    03BF           MOVW    W^PB+DIB$W_UNIT+8,-
               0342'CF           03C3                   W^MB_DEV_CHAR+DIB$W_UNIT        ; the unit # is a variable
                                 03C6                                                   ; and must be filled in
          56   036E'CF     DE    03C6           MOVAL   W^PB+8,R6                       ; set buffer address
          57   0336'CF     DE    03CB           MOVAL   W^MB_DEV_CHAR,R7                ; set good data address
               58    28    D0    03D0           MOVL    #MB_CHAR_SIZE,R8                ; set the byte count
                     00    DD    03D3           PUSHL   #0                              ; push expected IO status
          1287'CF    01    FB    03D5           CALLS   #1,W^BUF_CHECK                  ; check the resulting buffer
          56   03EA'CF     DE    03DA           MOVAL   W^SB+8,R6                       ; set buffer address
                     00    DD    03DF           PUSHL   #0                              ; push expected IO status
          1287'CF    01    FB    03E1           CALLS   #1,W^BUF_CHECK                  ; check the secondary buf
 035E'CF    00   036E'CF    00    2C    03E6           MOVC5   #0,W^PB+8,#0,W^PL,W^PB+8        ; init the buffers
                 036E'CF                03EF
 0362'CF    00   03EA'CF    00    2C    03F2           MOVC5   #0,W^SB+8,#0,W^SL,W^SB+8
                 03EA'CF                03FB
                                 03FE    ;+
                                 03FE    ;
                                 03FE    ; test _G form
                                 03FE    ;
                                 03FE    ;-
                                 03FE           NEXT_TEST
                                 03FE
                                 03FE    STP13:
          0004'CF    0D    D0    03FE                   MOVL    #13,W^CURRENT_TC
                     00    DD    0403                   PUSHL   #0
          1385'CF    01    FB    0405                   CALLS   #1,W^REG_SAVE
     00C5'CF   031E'CF     B0    040A           MOVW    W^MBCHAN,W^GETC+GETCHN$_CHAN    ; set the channel #
```

```
                                                                 F  4
SATSSS01                              - SATS SYSTEM SERVICE TESTS  (SUCC S.C.) 16-SEP-1984 00:44:47  VAX/VMS Macro V04-00        Page  18
V04-000                               GETCHN TESTS                              5-SEP-1984 04:29:37  [UETPSY.SRC]SATSSS01.MAR;1        (2)

                                         0411    632         $GETCHN_G W^GETC                          ; try _G form
                                         041A    633         FAIL_CHECK SS$_NORMAL                     ; check for success
                                01   DD  041A                         PUSHL    #SS$_NORMAL
                      138F'CF   01   FB  041C                         CALLS    #1,W^REG_CHECK
                                00   DD  0421    634         PUSHL    #0                               ; push expected IO status
                      1287'CF   01   FB  0423    635         CALLS    #1,W^BUF_CHECK                   ; check the returned buffer
                      56   036E'CF   DE  0428    636         MOVAL    W^PB+8,R6                        ; check the primary buffer
                                00   DD  042D    637         PUSHL    #0                               ; push expected IO status
                      1287'CF   01   FB  042F    638         CALLS    #1,W^BUF_CHECK                   ; for failures
     035E'CF     00   036E'CF   00   2C  0434    639         MOVC5    #0,W^PB+8,#0,W^PL,W^PB+8         ; init the buffers
                           036E'CF       043D
     0362'CF     00   03EA'CF   00   2C  0440    640         MOVC5    #0,W^SB+8,#0,W^SL,W^SB+8
                           03EA'CF       0449
```

```
                                044C   642          .SBTTL  GETDEV
                                044C   643 ;+
                                044C   644 ;
                                044C   645 ; $GETDEV tests
                                044C   646 ;
                                044C   647 ;
                                044C   648 ;-
                                044C   649          NEXT_TEST
                                044C
                                044C       STP14:
        0004'CF    0E    D0     044C                     MOVL    #14,W^CURRENT_TC
                   00    DD     0451                     PUSHL   #0
        1385'CF    01    FB     0453                     CALLS   #1,W^REG_SAVE
     0307'CF   0060'CF   DE     0458   650          MOVAL   W^GETDEV,W^SERV_NAME          ; set service name
     0159'CF   01FE'CF   DE     045F   651          MOVAL   W^UM,W^MODE                   ; set the mode
                                0466   652          $GETDEV_S DEVNAM=W^MBNAM,-
                                0466   653                  PRILEN=W^PL,-
                                0466   654                  PRIBUF=W^PB,-
                                0466   655                  SCDLEN=W^SL,-
                                0466   656                  SCDBUF=W^SB                   ; try the _S
                                0481   657          FAIL_CHECK SS$_NORMAL                 ; check success
                   01    DD     0481                     PUSHL   #SS$_NORMAL
        138F'CF    01    FB     0483                     CALLS   #1,W^REG_CHECK
                   00    DD     0488   658          PUSHL   #0                            ; push expected IO status
        1287'CF    01    FB     048A   659          CALLS   #1,W^BUF_CHECK                ; check the resulting buffer
        56   03EA'CF     DE     048F   660          MOVAL   W^SB+8,R6                     ; set buffer address
                   00    DD     0494   661          PUSHL   #0                            ; push expected IO status
        1287'CF    01    FB     0496   662          CALLS   #1,W^BUF_CHECK                ; check secondary buffer
 035E'CF    00   036E'CF   00    2C     049B   663          MOVC5   #0,W^PB+8,#0,W^PL,W^PB+8      ; init the buffers
                 036E'CF                04A4
 0362'CF    00   03EA'CF   00    2C     04A7   664          MOVC5   #0,W^SB+8,#0,W^SL,W^SB+8
                 03EA'CF                04B0
                                04B3   665 ;+
                                04B3   666 ;
                                04B3   667 ; test _G form
                                04B3   668 ;
                                04B3   669 ;-
                                04B3   670          NEXT_TEST
                                04B3
                                04B3       STP15:
        0004'CF    0F    D0     04B3                     MOVL    #15,W^CURRENT_TC
                   00    DD     04B8                     PUSHL   #0
        1385'CF    01    FB     04BA                     CALLS   #1,W^REG_SAVE
                                04BF   671          $GETDEV_G W^GETD                      ; try _G form
                                04C8   672          FAIL_CHECK SS$_NORMAL                 ; check for success
                   01    DD     04C8                     PUSHL   #SS$_NORMAL
        138F'CF    01    FB     04CA                     CALLS   #1,W^REG_CHECK
                   00    DD     04CF   673          PUSHL   #0                            ; push expected IO status
        1287'CF    01    FB     04D1   674          CALLS   #1,W^BUF_CHECK                ; check the returned buffer
        56   036E'CF     DE     04D6   675          MOVAL   W^PB+8,R6                     ; set the buffer address
                   00    DD     04DB   676          PUSHL   #0                            ; set expected IO status
        1287'CF    01    FB     04DD   677          CALLS   #1,W^BUF_CHECK                ; check the primary buffer
```

```
                                04E2   679              .SBTTL INPUT AND OUTPUT TESTS
                                04E2   680      ;+
                                04E2   681      ;
                                04E2   682      ; $INPUT and $OUTPUT tests
                                04E2   683      ;
                                04E2   684      ; try $OUTPUT with small transfer and a local EFN
                                04E2   685      ;
                                04E2   686      ;-
                                04E2   687              NEXT_TEST
                                04E2
                                04E2           STP16:
        0004'CF    10     D0    04E2                            MOVL    #16,W^CURRENT_TC
                   00     DD    04E7                            PUSHL   #0
        1385'CF    01     FB    04E9                            CALLS   #1,W^REG_SAVE
   0307'CF   0067'CF      DE    04EE   688              MOVAL   W^OUTPUT,W^SERV_NAME            ; set service name
   0159'CF   01FE'CF      DE    04F5   689              MOVAL   W^UM,W^MODE                     ; set the mode
                                04FC   690              $QIO_S  CHAN=W^MBCHAN,-
                                04FC   691                      FUNC=#IO$_READVBLK,-
                                04FC   692                      P1  =W^GETBUF+8,-
                                04FC   693                      P2  =#1                         ; let the output finish
                                051D   694              $OUTPUT CHAN=W^MBCHAN,-
                                051D   695                      LENGTH=#1,-
                                051D   696                      BUFFER=W^TEST_DATA,-
                                051D   697                      IOSB=W^STAT,-
                                051D   698                      EFN=#2                          ; try output,small, & local EFN
                                0540   699              FAIL_CHECK SS$_NORMAL                   ; check for success
                   01     DD    0540                            PUSHL   #SS$_NORMAL
        138F'CF    01     FB    0542                            CALLS   #1,W^REG_CHECK
        56    01DB'CF     DE    0547   700              MOVAL   W^GETBUF+8,R6                   ; set input address
        57    0250'CF     DE    054C   701              MOVAL   W^TEST_DATA,R7                  ; set good data address
              58   01     D0    0551   702              MOVL    #1,R8                           ; set the byte count
  0071'CF    00010001 8F  D0    0554   703              MOVL    #1@16!SS$_NORMAL,W^STAT1        ; set dummy status
             00010001 8F  DD    055D   704              PUSHL   #1@16!SS$_NORMAL                ; set expected IO status
        1287'CF    01     FB    0563   705              CALLS   #1,W^BUF_CHECK                  ; check the results
           01DB'CF        D4    0568   706              CLRL    W^GETBUF+8                      ; init the buffer
                                056C   707      ;+
                                056C   708      ;
                                056C   709      ; test $INPUT with small transfer and local EFN
                                056C   710      ;
                                056C   711      ;-
                                056C   712              NEXT_TEST
                                056C
                                056C           STP17:
        0004'CF    11     D0    056C                            MOVL    #17,W^CURRENT_TC
                   00     DD    0571                            PUSHL   #0
        1385'CF    01     FB    0573                            CALLS   #1,W^REG_SAVE
   0307'CF   0053'CF      DE    0578   713              MOVAL   W^INPUT,W^SERV_NAME             ; set service name
                                057F   714              $QIO_S  CHAN=W^MBCHAN,-
                                057F   715                      FUNC=#IO$_WRITEVBLK,-
                                057F   716                      P1  =W^TEST_DATA,-
                                057F   717                      P2  =#1                         ; put data there to read
                                059E   718              $INPUT  CHAN=W^MBCHAN,-
                                059E   719                      LENGTH=#1,-
                                059E   720                      BUFFER=W^GETBUF+8,-
                                059E   721                      IOSB=W^STAT,-
                                059E   722                      EFN=#2                          ; try input,small, & local EFN
                                05BF   723              FAIL_CHECK SS$_NORMAL                   ; check for success
```

```
                       01   DD  05BF                      PUSHL   #SS$_NORMAL
           138F'CF     01   FB  05C1                      CALLS   #1,W^REG_CHECK
            00010001   8F   DD  05C6              PUSHL   #1@16!SS$_NORMAL                       ; set expected IO status
           1287'CF     01   FB  05CC              CALLS #1,W^BUF_CHECK                           ; check transfered data
                 01DB'CF    D4  05D1              CLRL    W^GETBUF+8                             ; init the buffer
     58     00000084   8F   D0  05D5              MOVL    #132,R8                                ; set new byte count
                                05DC      ;+
                                05DC      ;
                                05DC      ; test $OUTPUT with large transfer and common EFN
                                05DC      ;
                                05DC      ;-
                                05DC              NEXT_TEST
                                05DC
                                05DC      STP18:
           0004'CF     12   D0  05DC                      MOVL    #18,W^CURRENT_TC
                       00   DD  05E1                      PUSHL   #0
           1385'CF     01   FB  05E3                      CALLS   #1,W^REG_SAVE
     0307'CF   0067'CF      DE  05E8              MOVAL   W^OUTPUT,W^SERV_NAME                   ; set service name
                                05EF              $ASCEFC_S #65,W^EFCNAM                         ; make EFN 65
                                0604              $QIO_S  CHAN=W^MBCHAN,-
                                0604                      FUNC=#IO$_READVBLK,-
                                0604                      P1  =W^GETBUF+8,-
                                0604                      P2  =#132                              ; let the $OUTPUT complete
                                0627              $OUTPUT CHAN=W^MBCHAN,-
                                0627                      LENGTH=#132,-
                                0627                      BUFFER=W^TEST_DATA,-
                                0627                      IOSB=W^STAT,-
                                0627                      EFN=#65                                ; try output, large with common EFN
                                0652              FAIL_CHECK SS$_NORMAL                          ; check for success
                       01   DD  0652                      PUSHL   #SS$_NORMAL
           138F'CF     01   FB  0654                      CALLS   #1,W^REG_CHECK
   0071'CF  00840001   8F   D0  0659              MOVL    #132@16!SS$_NORMAL,W^STAT1             ; set dummy status
            00840001   8F   DD  0662              PUSHL   #132@16!SS$_NORMAL                     ; set expected IO status
           1287'CF     01   FB  0668              CALLS   #1,W^BUF_CHECK                         ; check the buffer
 0084 8F    00   01DB'CF   00   2C  066D              MOVC5   #0,W^GETBUF+8,#0,#132,W^GETBUF+8 ; init the buffer
                 01DB'CF        0676
                                0679      ;+
                                0679      ;
                                0679      ; test $INPUT with large transfer and common EFN
                                0679      ;
                                0679      ;-
                                0679              NEXT_TEST
                                0679
                                0679      STP19:
           0004'CF     13   D0  0679                      MOVL    #19,W^CURRENT_TC
                       00   DD  067E                      PUSHL   #0
           1385'CF     01   FB  0680                      CALLS   #1,W^REG_SAVE
     0307'CF   0053'CF      DE  0685              MOVAL   W^INPUT,W^SERV_NAME                    ; set service name
                                068C              $QIO_S  CHAN=W^MBCHAN,-
                                068C                      FUNC=#IO$_WRITEVBLK,-
                                068C                      P1  =W^TEST_DATA,-
                                068C                      P2  =#132                              ; put data out to read
                                06AF              $INPUT  CHAN=W^MBCHAN,-
                                06AF                      LENGTH=#132,-
                                06AF                      BUFFER=W^GETBUF+8,-
                                06AF                      IOSB=W^STAT,-
                                06AF                      EFN=#65                                ; try input, large with common EFN
```

```
                                      06D8          FAIL_CHECK SS$_NORMAL                     ; check for success
                           01     DD  06D8                  PUSHL   #SS$_NORMAL
                  138F'CF  01     FB  06DA                  CALLS   #1,W^REG_CHECK
                00840001   8F     DD  06DF          PUSHL   #132@16!SS$_NORMAL                ; set expected IO status
                  1287'CF  01     FB  06E5          CALLS #1,W^BUF_CHECK                      ; check transfered data
  0084 8F     00  01DB'CF  00     2C  06EA          MOVC5   #0,W^GETBUF+8,#0,#132,W^GETBUF+8 ; init the buffer
                     01DB'CF          06F3
```

```
                             06F6          .SBTTL  QIO TESTS
                             06F6    ;+
                             06F6    ;
                             06F6    ; $QIO tests
                             06F6    ;
                             06F6    ; test local EFN = 3, IO$_WRITEVBLK, _S, 1 byte transfer
                             06F6    ;
                             06F6    ;-
           58     01    D0   06F6          MOVL    #1,R8                                   ; set byte count
                             06F9          NEXT_TEST
                             06F9
                             06F9    STP20:
      0004'CF     14    D0   06F9                    MOVL    #20,W^CURRENT_TC
                  00    DD   06FE                    PUSHL   #0
      1385'CF     01    FB   0700                    CALLS   #1,W^REG_SAVE
0307'CF   006E'CF       DE   0705          MOVAL   W^QIO,W^SERV_NAME                       ; set service name
                             070C          $QIO_S  EFN =#3,-
                             070C                  CHAN=W^MBCHAN,-
                             070C                  FUNC=#IO$_WRITEVBLK,-
                             070C                  IOSB=W^STAT,-
                             070C                  P1   =W^TEST_DATA,-
                             070C                  P2   =#1                             ; try _S local bc = 1 writevblk
                             072D          FAIL_CHECK SS$_NORMAL                           ; check success
                  01    DD   072D                    PUSHL   #SS$_NORMAL
      138F'CF     01    FB   072F                    CALLS   #1,W^REG_CHECK
                             0734    ;+
                             0734    ;
                             0734    ; test local EFN = 31, IO$_READVBLK, _G, 1 byte transfer
                             0734    ;
                             0734    ;-
                             0734          NEXT_TEST
                             0734
                             0734    STP21:
      0004'CF     15    D0   0734                    MOVL    #21,W^CURRENT_TC
                  00    DD   0739                    PUSHL   #0
      1385'CF     01    FB   073B                    CALLS   #1,W^REG_SAVE
         0105'CF        D4   0740          CLRL    W^QIOP+QIO$_ASTADR                      ; disable AST's
                             0744          $QIO_G  W^QIOP                                  ; try _G local bc = 1 readvblk
                             074D          FAIL_CHECK SS$_NORMAL                           ; check success
                  01    DD   074D                    PUSHL   #SS$_NORMAL
      138F'CF     01    FB   074F                    CALLS   #1,W^REG_CHECK
                             0754          $WAITFR_S EFN=#3                                ; wait for the writevblk
                             075D          $WAITFR_S EFN=#31                               ; wait for the readvblk
    00010001 8F         DD   0766          PUSHL   #1@16!SS$_NORMAL                        ; set expected IO status
      1287'CF     01    FB   076C          CALLS   #1,W^BUF_CHECK                          ; check the results
         01DB'CF        D4   0771          CLRL    W^GETBUF+8                              ; init the buffer
           58     02    D0   0775          MOVL    #2,R8                                   ; set byte count
                             0778    ;+
                             0778    ;
                             0778    ; test common EFN = 65, IO$_READLBLK, _S, 2 byte transfer
                             0778    ;
                             0778    ;-
                             0778          NEXT_TEST
                             0778
                             0778    STP22:
      0004'CF     16    D0   0778                    MOVL    #22,W^CURRENT_TC
                  00    DD   077D                    PUSHL   #0
```

```
              1385'CF   01   FB  077F                         CALLS   #1,W^REG_SAVE
                                 0784    810        $QIO_S  EFN=#65,-
                                 0784    811                CHAN=W^MBCHAN,-
                                 0784    812                FUNC=#IO$_READLBLK,-
                                 0784    813                IOSB=W^STAT,-
                                 0784    814                P1  =W^GETBUF+8,-
                                 0784    815                P2  =#2                       ; try common EFN READLBLK
                                 07A9    816        FAIL_CHECK SS$_NORMAL                 ; check success
                        01   DD  07A9                         PUSHL   #SS$_NORMAL
              138F'CF   01   FB  07AB                         CALLS   #1,W^REG_CHECK
                                 07B0    817 ;+
                                 07B0    818 ;
                                 07B0    819 ; test common EFN = 92, IO$_WRITELBLK, _G, 2 byte transfer
                                 07B0    820 ;
                                 07B0    821 ;-
                                 07B0    822        NEXT_TEST
                                 07B0
                                 07B0        STP23:
              0004'CF   17   D0  07B0                         MOVL    #23,W^CURRENT_TC
                        00   DD  07B5                         PUSHL   #0
              1385'CF   01   FB  07B7                         CALLS   #1,W^REG_SAVE
      00F5'CF   0000005C 8F   D0  07BC    823        MOVL    #92,W^QIOP+QIO$_EFN           ; set EFN
            00FD'CF   20   D0  07C5    824        MOVL    #IO$_WRITELBLK,W^QIOP+QIO$_FUNC ; set FUNC
       010D'CF   0250'CF   DE  07CA    825        MOVAL   W^TEST_DATA,W^QIOP+QIO$_P1     ; set transfer address
            0111'CF   02   D0  07D1    826        MOVL    #2,W^QIOP+QIO$_P2             ; set byte count
                                 07D6    827        $QIO_G  W^QIOP                        ; try common EFN writelblk
                                 07DF    828        FAIL_CHECK SS$_NORMAL                 ; check success
                        01   DD  07DF                         PUSHL   #SS$_NORMAL
              138F'CF   01   FB  07E1                         CALLS   #1,W^REG_CHECK
                                 07E6    829        $WAITFR_S EFN=#65                     ; wait for readlblk
                                 07F3    830        $WAITFR_S EFN=#92                     ; wait for writlblk
           00020001 8F   DD  0800    831        PUSHL   #2@16!SS$_NORMAL              ; set expected IO status
            1287'CF   01   FB  0806    832        CALLS   #1,W^BUF_CHECK                ; check transfer
                01DB'CF   D4  080B    833        CLRL    W^GETBUF+8                    ; init the buffer
       58   00000084 8F   D0  080F    834        MOVL    #132,R8                       ; set byte count
                                 0816    835 ;+
                                 0816    836 ;
                                 0816    837 ; test AST, IO$_WRITEPBLK, _S, 132 byte transfer
                                 0816    838 ;
                                 0816    839 ;-
                                 0816    840        NEXT_TEST
                                 0816
                                 0816        STP24:
              0004'CF   18   D0  0816                         MOVL    #24,W^CURRENT_TC
                        00   DD  081B                         PUSHL   #0
              1385'CF   01   FB  081D                         CALLS   #1,W^REG_SAVE
                                 0822    841        $QIO_S  CHAN=W^MBCHAN,-
                                 0822    842                FUNC=#IO$_WRITEPBLK,-
                                 0822    843                IOSB=W^STAT,-
                                 0822    844                ASTADR=W^AST1,-
                                 0822    845                ASTPRM=#1,-
                                 0822    846                P1  =W^TEST_DATA,-
                                 0822    847                P2  =#132                     ; try AST writepblk
                        50   DD  084B    848        PUSHL   R0                            ; save the QIO status
                                 084D    849        $SETAST_S ENBFLG=#0                   ; let things get checked
                     50 8ED0  0856    850        POPL    R0                            ; reset the QIO status
                                 0859    851                                              ; before the AST's start
                                 0859   852                                                     ; to fly!
                                 0859   853             FAIL_CHECK SS$_NORMAL                   ; check success
                      01    DD   0859                           PUSHL   #SS$_NORMAL
            138F'CF   01    FB   085B                           CALLS   #1,W^REG_CHECK
                                 0860   854 ;+
                                 0860   855 ;
                                 0860   856 ; test AST, IO$_READPBLK, _G, byte count 132
                                 0860   857 ;
                                 0860   858 ;-
                                 0860   859             NEXT_TEST
                                 0860
                                 0860       STP25:
            0004'CF   19    D0   0860                           MOVL    #25,W^CURRENT_TC
                      00    DD   0865                           PUSHL   #0
            1385'CF   01    FB   0867                           CALLS   #1,W^REG_SAVE
            00FD'CF   0C    D0   086C   860             MOVL    #IO$_READPBLK,W^QIOP+QIO$_FUNC  ; set FUNC
      0105'CF   08F0'CF     DE   0871   861             MOVAL   W^AST2,W^QIOP+QIO$_ASTADR       ; set ASTADR
            0109'CF   02    D0   0878   862             MOVL    #2,W^QIOP+QIO$_ASTPRM           ; set ASTPRM
      010D'CF   01DB'CF     DE   087D   863             MOVAL   W^GETBUF+8,W^QIOP+QIO$_P1       ; set read buffer adr
   0111'CF    00000084 8F   D0   0884   864             MOVL    #132,W^QIOP+QIO$_P2             ; set byte count
                                 088D   865             $QIO_G  W^QIOP                          ; try AST delivery _G
                                 0896   866             FAIL_CHECK SS$_NORMAL                   ; check success
                      01    DD   0896                           PUSHL   #SS$_NORMAL
            138F'CF   01    FB   0898                           CALLS   #1,W^REG_CHECK
                                 089D   867             $SETAST_S ENBFLG=#1                     ; let all heck break loose
                                 08A6   868             $WAITFR_S EFN=#92                       ; let the dust settle
           00840001 8F      DD   08B3   869             PUSHL   #132@16!SS$_NORMAL              ; set expected IO status
            1287'CF   01    FB   08B9   870             CALLS   #1,W^BUF_CHECK                  ; check transfer
 0084 8F    00   01DB'CF  00  2C  08BE   871             MOVC5   #0,W^GETBUF+8,#0,#132,W^GETBUF+8 ; init the buffer
                 01DB'CF         08C7
                   0046     31   08CA   872             BRW     NEXT                            ; skip over AST routines
                                 08CD   873 ;+
                                 08CD   874 ;
                                 08CD   875 ; service writelblk AST
                                 08CD   876 ;
                                 08CD   877 ;-
                                 08CD   878 AST1:
                           001C  08CD   879             .WORD   ^M<R2,R3,R4>
                                 08CF   880             NEXT_TEST
                                 08CF
                                 08CF       STP26:
            0004'CF   1A    D0   08CF                           MOVL    #26,W^CURRENT_TC
                      00    DD   08D4                           PUSHL   #0
            1385'CF   01    FB   08D6                           CALLS   #1,W^REG_SAVE
              01   04 AC    D1   08DB   881             CMPL    4(AP),#1                        ; right AST parameter?
                      0E    13   08DF   882             BEQL    10$                             ; br if yes
                   04 AC    DD   08E1   883             PUSHL   4(AP)                           ; push received
                      01    DD   08E4   884             PUSHL   #1                              ; push expected
                0193'CF     DF   08E6   885             PUSHAL  W^ASTEXP                        ; push string variable
            13D1'CF   03    FB   08EA   886             CALLS   #3,W^PRINT_FAIL                 ; print the failure
                                 08EF   887 10$:
                            04   08EF   888             RET                                     ; return
                                 08F0   889 ;+
                                 08F0   890 ;
                                 08F0   891 ; test the readlblk AST
                                 08F0   892 ;
                                 08F0   893 ;-
```

```
                         08F0     AST2:
                 001C    08F0             .WORD   ^M<R2,R3,R4>
                         08F2             NEXT_TEST
                         08F2
                         08F2     STP27:
  0004'CF    1B    D0    08F2                     MOVL    #27,W^CURRENT_TC
             00    DD    08F7                     PUSHL   #0
  1385'CF    01    FB    08F9                     CALLS   #1,W^REG_SAVE
   02    04 AC     D1    08FE             CMPL    4(AP),#2                         ; right AST parameter?
             0E    13    0902             BEQL    10$                              ; br if yes
          04 AC    DD    0904             PUSHL   4(AP)                            ; push received
             02    DD    0907             PUSHL   #2                               ; push expected
       0193'CF     DF    0909             PUSHAL  W^ASTEXP                         ; push string variable
  13D1'CF    03    FB    090D             CALLS   #3,W^PRINT_FAIL                  ; print the error
                         0912     10$:
                   04    0912             RET                                      ; return
                         0913     ;+
                         0913     ;
                         0913     ; test IO$_SETMODE, _S, READATTN
                         0913     ;
                         0913     ;-
                         0913     NEXT:
                         0913             NEXT_TEST
                         0913
                         0913     STP28:
  0004'CF    1C    D0    0913                     MOVL    #28,W^CURRENT_TC
             00    DD    0918                     PUSHL   #0
  1385'CF    01    FB    091A                     CALLS   #1,W^REG_SAVE
                         091F             $QIO_S  CHAN=W^MBCHAN,-
                         091F                     FUNC=#IO$_SETMODE!IO$M_READATTN,-
                         091F                     EFN =#2,-
                         091F                     P1  =W^AST3,-
                         091F                     P2  =#3,-
                         091F                     P3  =#PSL$C_USER                 ; try _S SETMODE
                         0942             FAIL_CHECK SS$_NORMAL                    ; check success
             01    DD    0942                     PUSHL   #SS$_NORMAL
  138F'CF    01    FB    0944                     CALLS   #1,W^REG_CHECK
                         0949             $WAITFR_S EFN=#2                         ; let it finish
     0105'CF       D4    0952             CLRL    W^QIOP+QIO$_ASTADR               ; disable AST's for this one
     0109'CF       D4    0956             CLRL    W^QIOP+QIO$_ASTPRM
                         095A             $SETAST_S ENBFLG=#0                      ; hold back on the reins
                         0963             $QIO_G  W^QIOP                           ; force the READATTN AST
                         096C             FAIL_CHECK SS$_NORMAL                    ; check success
             01    DD    096C                     PUSHL   #SS$_NORMAL
  138F'CF    01    FB    096E                     CALLS   #1,W^REG_CHECK
                         0973             $SETAST_S ENBFLG=#1                      ; let it fly
           0045    31    097C             BRW     NEXT1                            ; skip over AST routine
                         097F     ;+
                         097F     ;
                         097F     ; service READATTN AST
                         097F     ;
                         097F     ;-
                         097F     AST3:
                 0000    097F             .WORD   0
                         0981             NEXT_TEST
                         0981
                         0981     STP29:
```

```
           0004'CF    1D   D0  0981                      MOVL    #29,W^CURRENT_TC
                      00   DD  0986                      PUSHL   #0
           1385'CF    01   FB  0988                      CALLS   #1,W^REG_SAVE
            03   04 AC     D1  098D   935      CMPL    4(AP),#3                        ; correct AST?
                      0E   13  0991   936      BEQL    10$                             ; br if OK
                 04 AC     DD  0993   937      PUSHL   4(AP)                           ; push received
                      03   DD  0996   938      PUSHL   #3                              ; push expected
              0193'CF      DF  0998   939      PUSHAL  W^ASTEXP                        ; push the string variable
           13D1'CF    03   FB  099C   940      CALLS   #3,W^PRINT_FAIL                 ; print the failure
                               09A1   941 10$:
           00FD'CF    30   D0  09A1   942      MOVL    #IO$_WRITEVBLK,W^QIOP+QIO$_FUNC ; set the new mode
                               09A6   943      $QIO_G  W^QIOP                          ; and eat the read pending
                               09AF   944      FAIL_CHECK SS$_NORMAL                   ; check success
                      01   DD  09AF                      PUSHL   #SS$_NORMAL
           138F'CF    01   FB  09B1                      CALLS   #1,W^REG_CHECK
                               09B6   945      $WAITFR_S EFN=#92                       ; wait for it to digest.
                          04   09C3   946      RET                                     ; carry on
                               09C4   947 ;+
                               09C4   948 ;
                               09C4   949 ; test IO$_SETMODE, _G, WRTATTN
                               09C4   950 ;
                               09C4   951 ;-
                               09C4   952 NEXT1:
                               09C4   953      NEXT_TEST
                               09C4
                               09C4            STP30:
           0004'CF    1E   D0  09C4                      MOVL    #30,W^CURRENT_TC
                      00   DD  09C9                      PUSHL   #0
           1385'CF    01   FB  09CB                      CALLS   #1,W^REG_SAVE
          00000123 8F      D0  09D0   954      MOVL    #IO$_SETMODE!IO$M_WRTATTN,-
               00FD'CF         09D6   955              W^QIOP+QIO$_FUNC                ; set new function
   010D'CF    0A3A'CF      DE  09D9   956      MOVAL   W^AST4,W^QIOP+QIO$_P1           ; set new P1
           0111'CF    04   D0  09E0   957      MOVL    #4,W^QIOP+QIO$_P2               ; set new P2
           0115'CF    03   D0  09E5   958      MOVL    #PSL$C_USER,W^QIOP+QIO$_P3      ; set new P3
                               09EA   959      $QIO_G  W^QIOP                          ; try _G setmode
                               09F3   960      FAIL_CHECK SS$_NORMAL                   ; check success
                      01   DD  09F3                      PUSHL   #SS$_NORMAL
           138F'CF    01   FB  09F5                      CALLS   #1,W^REG_CHECK
                               09FA   961      $WAITFR_S EFN=#92                       ; wait for it to complete
                               0A07   962      $SETAST_S ENBFLG=#0                     ; hold back on the reins
       000000FD'EF    30   D0  0A10   963      MOVL    #IO$_WRITEVBLK,QIOP+QIO$_FUNC   ; set new function
   010D'CF    0250'CF      DE  0A17   964      MOVAL   W^TEST_DATA,W^QIOP+QIO$_P1      ; set new P1
                               0A1E   965      $QIO_G  W^QIOP                          ; kick off WRTATTN AST
                               0A27   966      FAIL_CHECK SS$_NORMAL                   ; check success
                      01   DD  0A27                      PUSHL   #SS$_NORMAL
           138F'CF    01   FB  0A29                      CALLS   #1,W^REG_CHECK
                               0A2E   967      $SETAST_S ENBFLG=#1                     ; let it fly
                 004C     31   0A37   968      BRW     NEXT2                           ; skip AST routine
                               0A3A   969 ;+
                               0A3A   970 ;
                               0A3A   971 ; service WRTATTN AST
                               0A3A   972 ;
                               0A3A   973 ;-
                               0A3A   974 AST4:
                        0000   0A3A   975      .WORD   0
                               0A3C   976      NEXT_TEST
                               0A3C
```

```
                            0A3C        STP31:
         0004'CF    1F   D0 0A3C                        MOVL    #31,W^CURRENT_TC
                    00   DD 0A41                        PUSHL   #0
         1385'CF    01   FB 0A43                        CALLS   #1,W^REG_SAVE
           04    04 AC   D1 0A48              CMPL    4(AP),#4                        ; is it the right one?
                    0E   13 0A4C              BEQL    10$                             ; br if it's OK
                 04 AC   DD 0A4E              PUSHL   4(AP)                           ; save received
                    04   DD 0A51              PUSHL   #4                              ; save expected
            0193'CF      DF 0A53              PUSHAL  W^ASTEXP                        ; save string variable
         13D1'CF    03   FB 0A57              CALLS   #3,W^PRINT_FAIL                 ; print the error
                            0A5C        10$:
         00FD'CF    31   D0 0A5C              MOVL    #IO$_READVBLK,W^QIOP+QIO$_FUNC  ; set new function code
  010D'CF    01D8'CF     DE 0A61              MOVAL   W^GETBUF+8,W^QIOP+QIO$_P1       ; set new read address
                            0A68              $QIO_G  W^QIOP                          ; eat the write pending
                            0A71              FAIL_CHECK SS$_NORMAL                   ; check for success
                    01   DD 0A71                        PUSHL   #SS$_NORMAL
         138F'CF    01   FB 0A73                        CALLS   #1,W^REG_CHECK
                            0A78              $WAITFR_S EFN=#92                       ; and wait for it to digest
                         04 0A85              RET                                     ; bail out
                            0A86        ;+
                            0A86        ;
                            0A86        ; test IO$_SETCHAR, _S
                            0A86        ;
                            0A86        ; This function is not tested because of the lack of a device that is
                            0A86        ; allocatable and char. setable on the minimum configuration.
                            0A86        ;
                            0A86        ;-
                            0A86        NEXT2:
                                0A86  ;+
                                0A86  ;
                                0A86  ; test IO$_WRITEOF, _G
                                0A86  ;
                                0A86  ;-
                                0A86           NEXT_TEST
                                0A86
                                0A86  STP32:
           0004'CF   20    D0   0A86                     MOVL    #32,W^CURRENT_TC
                     00    DD   0A8B                     PUSHL   #0
           1385'CF   01    FB   0A8D                     CALLS   #1,W^REG_SAVE
                                0A92           $QIO_S  CHAN=W^MBCHAN,-
                                0A92                   FUNC=#IO$_WRITEOF,-
                                0A92                   EFN =#10                           ; issue the WRITEOF
                                0AAF           FAIL_CHECK SS$_NORMAL                      ; check success
                     01    DD   0AAF                   PUSHL   #SS$_NORMAL
           138F'CF   01    FB   0AB1                   CALLS   #1,W^REG_CHECK
   00F9'CF    031E'CF   3C   0AB6           MOVZWL  W^MBCHAN,W^QIOP+QIO$_CHAN          ; reset the channel
     00FD'CF   31    D0   0ABD           MOVL    #IO$_READVBLK,W^QIOP+QIO$_FUNC    ; set for the read
   010D'CF    01DB'CF   DE   0AC2           MOVAL   W^GETBUF+8,W^QIOP+QIO$_P1          ; set dummy address
  00000111'EF   02    D0   0AC9           MOVL    #2,QIOP+QIO$_P2                    ; set any byte count
                                0AD0           $QIO_G  W^QIOP                             ; issue a read
                                0AD9           FAIL_CHECK SS$_NORMAL                      ; check success
                     01    DD   0AD9                   PUSHL   #SS$_NORMAL
           138F'CF   01    FB   0ADB                   CALLS   #1,W^REG_CHECK
                                0AE0           $WAITFR_S EFN=#92                          ; wait for completion
  00000870 8F    0071'CF   D1   0AED           CMPL    W^STAT1,#SS$_ENDOFFILE             ; right status code?
                     13    13   0AF6           BEQL    10$                                ; br if OK
             0071'CF       DD   0AF8           PUSHL   W^STAT1                            ; push received
          00000870 8F      DD   0AFC           PUSHL   #SS$_ENDOFFILE                     ; push expected
             0182'CF       DF   0B02           PUSHAL  W^IOEXP                            ; push string variable
           13D1'CF   03    FB   0B06           CALLS   #3,W^PRINT_FAIL                    ; print the failure
                                0B0B  10$:
                                0B0B  ;+
                                0B0B  ;
                                0B0B  ; test IO$_ACCESS, _G
                                0B0B  ;
                                0B0B  ; Start testing disk files.  We first want to find the FID of [SYSTEST],
                                0B0B  ; which may be in a top level system directory.  Save that FID as the DID
                                0B0B  ; for further testing.
                                0B0B  ;
                                0B0B  ;-
                                0B0B           NEXT_TEST
                                0B0B
                                0B0B  STP33:
           0004'CF   21    D0   0B0B                     MOVL    #33,W^CURRENT_TC
                     00    DD   0B10                     PUSHL   #0
           1385'CF   01    FB   0B12                     CALLS   #1,W^REG_SAVE
                                0B17           $ASSIGN_S W^DISK,W^CHAN1                   ; assign the disk channel
                                0B28           $TRNLOG_S LOGNAM = W^TOPSYS,-              ; See if there is...
                                0B28                     RSLLEN = W^TOPSYS_DIR,-          ; ...a top level...
                                0B28                     RSLBUF = W^TOPSYS_DIR,-          ; ...system directory,...
                                0B28                     DSBMSK = #6                      ; ...defined system-wide
           50    0629 8F   B1   0B41           CMPW    #SS$_NOTRAN,R0                     ; If there's no translation...
                     72    13   0B46           BEQL    10$                                ; ...
             04E4'CF       B5   0B48           TSTW    W^TOPSYS_DIR                       ; ...or the trans is null...
                     6C    13   0B4C           BEQL    10$                                ; ...we have no top level dirs
```

```
                  007C 8F   BB  0B4E          PUSHR   #^M<R2,R3,R4,R5,R6>              ; Save these over MOVC, etc.
           56     04E4'CF   3C  0B52          MOVZWL  W^TOPSYS_DIR,R6                  ; Get top level dir name length
      04CC'CF     04C4'CF   28  0B57          MOVC3   W^DOT_DIR_SEMI,W^DOT_DIR_SEMI+8,- ; Form a file spec for...
                  04EC'C6       0B5E                  TOPSYS_DIR+8(R6)                 ; ...the dir name...
      04E4'CF     04C4'CF   A0  0B61          ADDW2   W^DOT_DIR_SEMI,W^TOPSYS_DIR      ; ...
                  007C 8F   BA  0B68          POPR    #^M<R2,R3,R4,R5,R6>              ; Clean up after MOVC, etc.
                                0B6C          $QIOW_S EFN=#16,-                        ; Get the top level...
                                0B6C                  CHAN=W^CHAN1,-                   ; ...system directory FID
                                0B6C                  FUNC=#IO$_ACCESS,-
                                0B6C                  IOSB=W^STAT,-
                                0B6C                  P1  =W^FIBDES,-
                                0B6C                  P2  =#TOPSYS_DIR,-
                                0B6C                  P5  =#ATR
                                0B97          FAIL_CHECK SS$_NORMAL                    ; Check success of call...
                       01   DD  0B97                  PUSHL   #SS$_NORMAL
            138F'CF    01   FB  0B99                  CALLS   #1,W^REG_CHECK
            0069'CF    01   D1  0B9E          CMPL    #SS$_NORMAL,W^STAT               ; ...and its results
                       5E   12  0BA3          BNEQ    20$                              ; BR if error occurred
                       3C   BB  0BA5          PUSHR   #^M<R2,R3,R4,R5>                 ; Save these over MOVC, etc.
    0470'CF     046A'CF    06   28  0BA7          MOVC3   #6,W^FIB+FIB$W_FID,W^FIB+FIB$W_DID ; Get the new DID...
046A'CF   06    00     00 8F    00   2C  0BAF          MOVC5   #0,#0,#0,#6,W^FIB+FIB$W_FID      ; ...and reset the FID
                       3C   BA  0BB8          POPR    #^M<R2,R3,R4,R5>                 ; Restore after MOVC, etc.
                                0BBA    10$:
      0493'CF     04AF'CF   DE  0BBA          MOVAL   W^SYSTEST_DIR,W^ATR+4            ; Point to SYSTEST dir name
                                0BC1          $QIO_S  EFN=#16,-
                                0BC1                  CHAN=W^CHAN1,-
                                0BC1                  FUNC=#IO$_ACCESS,-
                                0BC1                  IOSB=W^STAT,-
                                0BC1                  P1  =W^FIBDES,-
                                0BC1                  P2  =#SYSTEST_DIR,-
                                0BC1                  P5  =#ATR                        ; access file to get DID
                                0BEC          FAIL_CHECK SS$_NORMAL                    ; check success
                       01   DD  0BEC                  PUSHL   #SS$_NORMAL
            138F'CF    01   FB  0BEE                  CALLS   #1,W^REG_CHECK
                                0BF3          $WAITFR_S EFN=#16                        ; wait for completion
           01     0069'CF   D1  0BFC          CMPL    W^STAT,#SS$_NORMAL               ; check IO status
                       0F   13  0C01          BEQL    30$                              ; br if no error
                                0C03    20$:
                  0069'CF   DD  0C03          PUSHL   W^STAT                           ; push recieved
                       01   DD  0C07          PUSHL   #SS$_NORMAL                      ; push expected
                  0182'CF   DF  0C09          PUSHAL  W^IOEXP                          ; push string variable
            13D1'CF    03   FB  0C0D          CALLS   #3,W^PRINT_FAIL                  ; print the failure
                                0C12    30$:
    0470'CF     046A'CF    06   28  0C12          MOVC3   #6,W^FIB+FIB$W_FID,W^FIB+FIB$W_DID ; get the new DID
                                0C1A    ;+
                                0C1A    ;
                                0C1A    ; test IO$_CREATE, _S
                                0C1A    ;
                                0C1A    ; After ensuring that we have SYSPRV, set up access control and extension
                                0C1A    ; control.  Set up a test file, superseding any old one which may be present.
                                0C1A    ;
                                0C1A    ;-
                                0C1A          NEXT_TEST
                                0C1A
                                0C1A    STP34:
            0004'CF    22   D0  0C1A                  MOVL    #34,W^CURRENT_TC
                       00   DD  0C1F                  PUSHL   #0
```

```
        1385'CF    01    FB  0C21                        CALLS   #1,W^REG_SAVE
                             0C26              MODE      TO,10$,KRNL,NOREGS                 ; kernal mode to access PHD
     59   00000000'9F    D0  0C43              MOVL      @#CTL$GL_PHD,R9                    ; get process header address
        0051'CF    69    DE  0C4A              MOVAL     PHD$Q_PRIVMSK(R9),W^PRIVMASK       ; get priv mask address
                             0C4F              MODE      FROM,10$                           ; get back to user mode
                             0C50              PRIV      ADD,SYSPRV                         ; add SYSPRV priv.
             046A'CF     D4  0C70              CLRL      W^FIB+FIB$W_FID                    ; clear out the FID
             046E'CF     B4  0C74              CLRW      W^FIB+FIB$W_FID_RVN
 0466'CF   00000501 8F   D0  0C78              MOVL      #FIB$M_WRITE!FIB$M_NOREAD!-
                             0C81                         FIB$M_NOWRITE,W^FIB+FIB$L_ACCTL ; set new ACCTL
     047C'CF   0085 8F   B0  0C81              MOVW      #FIB$M_EXTEND!FIB$M_ALCON!-
                             0C88                         FIB$M_FILCON,W^FIB+FIB$W_EXCTL ; set new EXCTL
     047A'CF   0400 8F   B0  0C88              MOVW      #FIB$M_SUPERSEDE,W^FIB+FIB$W_NMCTL ; on top of file if there
        047E'CF    0F    D0  0C8F              MOVL      #15,W^FIB+FIB$L_EXSZ               ; set extend size to 15
                   00    DD  0C94              PUSHL     #0                                 ; push a dummy parameter
        1385'CF    01    FB  0C96              CALLS     #1,W^REG_SAVE                      ; save a register snapshot
                             0C9B              $QIO_S     EFN = #6,-
                             0C9B                        CHAN = W^CHAN1,-
                             0C9B                        FUNC = #IO$_CREATE!IO$M_CREATE!IO$M_ACCESS,-
                             0C9B                        IOSB = W^STAT,-
                             0C9B                          P1 = W^FIBDES,-
                             0C9B                          P2 = #FILENAME                  ; create the file
                             0CC2              FAIL_CHECK SS$_NORMAL                        ; check for success
                   01    DD  0CC2                        PUSHL   #SS$_NORMAL
        138F'CF    01    FB  0CC4                        CALLS   #1,W^REG_CHECK
                             0CC9              $WAITFR_S EFN=#6                             ; wait until done
     0F   006D'CF        D1  0CD2              CMPL      W^STAT+4,#15                       ; was it extended?
                   0F    18  0CD7              BGEQ      20$                                ; br if OK
        006D'CF          DD  0CD9              PUSHL     W^STAT+4                           ; push received
                   0F    DD  0CDD              PUSHL     #15                                ; push expected
        01A5'CF          DF  0CDF              PUSHAL    W^DISALL                           ; push string variable
        13D1'CF    03    FB  0CE3              CALLS     #3,W^PRINT_FAIL                    ; print the failure
                             0CE8      20$:
     01   0069'CF        D1  0CE8              CMPL      W^STAT,#SS$_NORMAL                 ; check the IO status
                   0F    13  0CED              BEQL      25$                                ; br if no errors
        0069'CF          DD  0CEF              PUSHL     W^STAT                             ; push recieved
                   01    DD  0CF3              PUSHL     #SS$_NORMAL                        ; push expected
        0182'CF          DF  0CF5              PUSHAL    W^IOEXP                            ; push string variable
        13D1'CF    03    FB  0CF9              CALLS     #3,W^PRINT_FAIL                    ; print the failure
                             0CFE      25$:
                             0CFE      ;+
                             0CFE      ;
                             0CFE      ; test IO$_MODIFY, _S
                             0CFE      ;
                             0CFE      ; Specify that our test file need not be contiguous and extend it by an
                             0CFE      ; amount equal to its original size.  Check that we've successfully modified
                             0CFE      ; the file.
                             0CFE      ;
                             0CFE      ;-
                             0CFE              NEXT_TEST
                             0CFE
                             0CFE      STP35:
        0004'CF    23    D0  0CFE                        MOVL    #35,W^CURRENT_TC
                   00    DD  0D03                        PUSHL   #0
        1385'CF    01    FB  0D05                        CALLS   #1,W^REG_SAVE
        047C'CF    04    AA  0D0A              BICW2     #FIB$M_FILCON,W^FIB+FIB$W_EXCTL ; remove contiguous mark
             0482'CF     D4  0D0F              CLRL      W^FIB+FIB$L_EXVBN                  ; allow the modify to work
```

```
                            0D13  1141          $QIO_S  EFN=#7,-
                            0D13  1142                  CHAN=W^CHAN1,-
                            0D13  1143                  FUNC=#IO$_MODIFY,-
                            0D13  1144                  IOSB=W^STAT,-
                            0D13  1145                  P1  =W^FIBDES,-
                            0D13  1146                  P2  =#FILENAME                   ; try to truncate with IO$_MODIFY
                            0D38  1147          FAIL_CHECK SS$_NORMAL
                01      DD  0D38                        PUSHL   #SS$_NORMAL
     138F'CF    01      FB  0D3A                        CALLS   #1,W^REG_CHECK
                            0D3F  1148          $WAITFR_S EFN=#7                         ; wait for completion
  01   0069'CF          D1  0D48  1149          CMPL    W^STAT,#SS$_NORMAL               ; check IO status
                0F      13  0D4D  1150          BEQL    10$                              ; br if no error
       0069'CF          DD  0D4F  1151          PUSHL   W^STAT                           ; push recieved
                01      DD  0D53  1152          PUSHL   #SS$_NORMAL                      ; push expected
       0182'CF          DF  0D55  1153          PUSHAL  W^IOEXP                          ; push string variable
     13D1'CF    03      FB  0D59  1154          CALLS   #3,W^PRINT_FAIL                  ; print the failure
                            0D5E  1155  10$:
     047C'CF    04      A8  0D5E  1156          BISW2   #FIB$M_FILCON,W^FIB+FIB$W_EXCTL  ; set a value to be over written
                            0D63  1157          $QIO_S  EFN=#5,-
                            0D63  1158                  CHAN=W^CHAN1,-
                            0D63  1159                  FUNC=#IO$_ACCESS,-
                            0D63  1160                  IOSB=W^STAT,-
                            0D63  1161                  P1  =W^FIBDES,-
                            0D63  1162                  P2  =#FILENAME
                            0D88  1163          FAIL_CHECK SS$_NORMAL                    ; check for success
                01      DD  0D88                        PUSHL   #SS$_NORMAL
     138F'CF    01      FB  0D8A                        CALLS   #1,W^REG_CHECK
                            0D8F  1164          $WAITFR_S EFN=#5                         ; wait for completion
  09 047C'CF    04      E1  0D98  1165          BBC     #FIB$M_FILCON,W^FIB+FIB$W_EXCTL,20$ ; if cleared then OK
       01[illegible]B'CF          DF  0D9E  1166          PUSHAL W^FILNOTMOD                      ; push string variable
     13D1'CF    01      FB  0DA2  1167          CALLS   #1,W^PRINT_FAIL                  ; print the failure
                            0DA7  1168  20$:
                            0DA7  1169  ;+
                            0DA7  1170  ;
                            0DA7  1171  ; Check that we may read and write the file with IO$_WRITEVBLK & IO$_READVBLK.
                            0DA7  1172  ;
                            0DA7  1173  ;-
                            0DA7  1174          NEXT_TEST
                            0DA7
                            0DA7        STP36:
     0004'CF    24      D0  0DA7                        MOVL    #36,W^CURRENT_TC
                00      DD  0DAC                        PUSHL   #0
     1385'CF    01      FB  0DAE                        CALLS   #1,W^REG_SAVE
       0069'CF          D4  0DB3  1175          CLRL    W^STAT                           ; clean the IO status blk
       0071'CF          D4  0DB7  1176          CLRL    W^STAT1
                            0DBB  1177          $QIO_S  EFN =#9,-
                            0DBB  1178                  CHAN=W^CHAN1,-
                            0DBB  1179                  FUNC=#IO$_WRITEVBLK,-
                            0DBB  1180                  IOSB=W^STAT,-
                            0DBB  1181                  P1  =W^TEST_DATA,-
                            0DBB  1182                  P2  =#132,-
                            0DBB  1183                  P3  =#1                          ; write 132 bytes to VBN 1
                            0DE2  1184          FAIL_CHECK SS$_NORMAL                    ; check success
                01      DD  0DE2                        PUSHL   #SS$_NORMAL
     138F'CF    01      FB  0DE4                        CALLS   #1,W^REG_CHECK
                            0DE9  1185          $WAITFR_S EFN=#9                         ; wait here til done
                            0DF2  1186          $QIO_S  EFN =#10,-
```

```
                           0DF2                      CHAN=W^CHAN1,-
                           0DF2                      FUNC=#IO$_READVBLK,-
                           0DF2                      IOSB=W^STAT1,-
                           0DF2                      P1  =W^GETBUF+8,-
                           0DF2                      P2  =#132,-
                           0DF2                      P3  =#1                          ; read 132 bytes from VBN 1
                           0E19            FAIL_CHECK SS$_NORMAL                     ; check success
                   01  DD  0E19                      PUSHL   #SS$_NORMAL
         138F'CF   01  FB  0E1B                      CALLS   #1,W^REG_CHECK
                           0E20            $WAITFR_S EFN=#10                         ; wait here til done
      56  01DB'CF      DE  0E29            MOVAL   W^GETBUF+8,R6                     ; set buffer address
      57  0250'CF      DE  0E2E            MOVAL   W^TEST_DATA,R7                    ; set good data address
   58  00000084  8F    D0  0E33            MOVL    #132,R8                           ; set byte count
       00840001  8F    DD  0E3A            PUSHL   #132@16!SS$_NORMAL                ; push expected status return
         11CB'CF   01  FB  0E40            CALLS   #1,W^DISK_BUF_CHECK               ; check the transfer
                           0E45  ;+
                           0E45  ;
                           0E45  ; test IO$_DEACCESS, _S
                           0E45  ;
                           0E45  ;-
                           0E45            NEXT_TEST
                           0E45
                           0E45  STP37:
         0004'CF   25  D0  0E45                      MOVL    #37,W^CURRENT_TC
                   00  DD  0E4A                      PUSHL   #0
         1385'CF   01  FB  0E4C                      CALLS   #1,W^REG_SAVE
           0069'CF     D4  0E51            CLRL    W^STAT                            ; clear IO status blks
           0071'CF     D4  0E55            CLRL    W^STAT1
18   00  01D3'CF   00  2C  0E59            MOVC5   #0,W^GETBUF,#0,#FIB$L_LOC_ADDR-
           0476'CF         0E60
                           0E63                    -FIB$L_WCC,W^FIB+FIB$L_WCC        ; clear unneeded stuff in FIB
                   00  DD  0E63            PUSHL   #0                                ; push a dummy parameter
         1385'CF   01  FB  0E65            CALLS   #1,W^REG_SAVE                     ; save a snapshot of regs
                           0E6A            $QIO_S  EFN =#5,-
                           0E6A                    CHAN=W^CHAN1,-
                           0E6A                    FUNC=#IO$_DEACCESS,-
                           0E6A                    IOSB=W^STAT1,-
                           0E6A                    P5  =#ATR,-
                           0E6A                    P1  =W^FIBDES                     ; try _S deaccess
                           0E91            FAIL_CHECK SS$_NORMAL                     ; check success
                   01  DD  0E91                      PUSHL   #SS$_NORMAL
         138F'CF   01  FB  0E93                      CALLS   #1,W^REG_CHECK
                           0E98            $WAITFR_S EFN=#5                          ; wait for completion
         0071'CF   01  D1  0EA1            CMPL    #SS$_NORMAL,W^STAT1               ; check IO status
                   0F  13  0EA6            BEQL    10$                               ; br if OK
           0071'CF     DD  0EA8            PUSHL   W^STAT1                           ; push recieved
                   01  DD  0EAC            PUSHL   #SS$_NORMAL                       ; push expected
           0182'CF     DF  0EAE            PUSHAL  W^IOEXP                           ; push string variable
         13D1'CF   03  FB  0EB2            CALLS   #3,W^PRINT_FAIL                   ; print the failure
                           0EB7  10$:
                           0EB7  ;+
                           0EB7  ;
                           0EB7  ; test IO$_DELETE, _S
                           0EB7  ;
                           0EB7  ;-
                           0EB7            NEXT_TEST
                           0EB7
```

```
                         0EB7        STP38:
   0004'CF    26    D0   0EB7                          MOVL    #38,W^CURRENT_TC
              00    DD   0EBC                          PUSHL   #0
   1385'CF    01    FB   0EBE                          CALLS   #1,W^REG_SAVE
      0069'CF       D4   0EC3   1233           CLRL    W^STAT                          ; init IO status
                         0EC7   1234           $QIO_S  EFN =#11,-
                         0EC7   1235                   CHAN=W^CHAN1,-
                         0EC7   1236                   FUNC=#IO$_DELETE!IO$M_DELETE,-
                         0EC7   1237                   IOSB=W^STAT,-
                         0EC7   1238                   P1  =W^FIBDES,-
                         0EC7   1239                   P2  =#FILENAME                  ; delete the file
                         0EEE   1240           FAIL_CHECK SS$_NORMAL                   ; check for success
              01    DD   0EEE                          PUSHL   #SS$_NORMAL
   138F'CF    01    FB   0EF0                          CALLS   #1,W^REG_CHECK
                         0EF5   1241           $WAITFR_S EFN=#11                       ; wait for completion
   0069'CF    01    D1   0EFE   1242           CMPL    #SS$_NORMAL,W^STAT              ; check IO status
              0F    13   0F03   1243           BEQL    10$                             ; br if OK
      0069'CF       DD   0F05   1244           PUSHL   W^STAT                          ; push recieved
              01    DD   0F09   1245           PUSHL   #SS$_NORMAL                     ; push expected
      0182'CF       DF   0F0B   1246           PUSHAL  W^IOEXP                         ; push string variable
   13D1'CF    03    FB   0F0F   1247           CALLS   #3,W^PRINT_FAIL                 ; print the failure
                         0F14   1248   10$:
                         0F14   1249           $DASSGN_S CHAN=W^CHAN1                  ; deassign the disk
```

```
                                 0F20          .SBTTL  QIOW TESTS
                                 0F20  ;+
                                 0F20  ;
                                 0F20  ; $QIOW tests
                                 0F20  ;
                                 0F20  ; The $QIO tests check most of the functionality of the QIO services.
                                 0F20  ; The purpose of these tests is to check the differences between
                                 0F20  ; $QIO and $QIOW.
                                 0F20  ;
                                 0F20  ; test _S and local EFN
                                 0F20  ;
                                 0F20  ;-
                                 0F20          NEXT_TEST
                                 0F20
                                 0F20  STP39:
          0004'CF    27    D0    0F20                  MOVL    #39,W^CURRENT_TC
                     00    DD    0F25                  PUSHL   #0
          1385'CF    01    FB    0F27                  CALLS   #1,W^REG_SAVE
     0307'CF    0072'CF    DE    0F2C          MOVAL W^QIOW,W^SERV_NAME                ; set service name
                                 0F33          $QIO_S  CHAN=W^MBCHAN,-
                                 0F33                  FUNC=#IO$_READVBLK,-
                                 0F33                  P1  =W^GETBUF+8,-
                                 0F33                  P2  =#80                        ; set up the mailbox
                                 0F56          $QIOW_S EFN =#16,-
                                 0F56                  CHAN=W^MBCHAN,-
                                 0F56                  FUNC=#IO$_WRITEVBLK,-
                                 0F56                  IOSB=W^STAT,-
                                 0F56                  P1  =W^TEST_DATA,-
                                 0F56                  P2  =#80                        ; try _S with local EFN
                                 0F7B          FAIL_CHECK SS$_NORMAL                   ; check for success
                     01    DD    0F7B                  PUSHL   #SS$_NORMAL
          138F'CF    01    FB    0F7D                  CALLS   #1,W^REG_CHECK
          56    01DB'CF    DE    0F82          MOVAL   W^GETBUF+8,R6                   ; set buffer address
          57    0250'CF    DE    0F87          MOVAL   W^TEST_DATA,R7                  ; set good data address
       58    00000050 8F   D0    0F8C          MOVL    #80,R8                          ; set the byte count
  0071'CF    00500001 8F   D0    0F93          MOVL    #80@16!SS$_NORMAL,W^STAT1       ; set dummy status
             00500001 8F   DD    0F9C          PUSHL   #80@16!SS$_NORMAL               ; set expected IO status
          1287'CF    01    FB    0FA2          CALLS   #1,W^BUF_CHECK                  ; check the data
0050 8F   00    01DB'CF    00    2C    0FA7    MOVC5   #0,W^GETBUF+8,#0,#80,W^GETBUF+8 ; init the buffer
                01DB'CF          0FB0
                                 0FB3  ;+
                                 0FB3  ;
                                 0FB3  ; test _G with local EFN
                                 0FB3  ;
                                 0FB3  ;-
                                 0FB3          NEXT_TEST
                                 0FB3
                                 0FB3  STP40:
          0004'CF    28    D0    0FB3                  MOVL    #40,W^CURRENT_TC
                     00    DD    0FB8                  PUSHL   #0
          1385'CF    01    FB    0FBA                  CALLS   #1,W^REG_SAVE
     012D'CF    031E'CF    D0    0FBF          MOVL    W^MBCHAN,W^QIOWP+QIOW$_CHAN     ; set the channel number
                                 0FC6          $QIO_S  CHAN=W^MBCHAN,-
                                 0FC6                  FUNC=#IO$_WRITEVBLK,-
                                 0FC6                  P1  =W^TEST_DATA,-
                                 0FC6                  P2  =#80                        ; set up the mailbox
                                 0FE9          $QIOW_G W^QIOWP                         ; try _G with local EFN
```

```
                                    0FF2  1295          FAIL_CHECK SS$_NORMAL                        ; check for success
                         01   DD    0FF2                          PUSHL   #SS$_NORMAL
                138F'CF  01   FB    0FF4                          CALLS   #1,W^REG_CHECK
               00500001  8F   DD    0FF9  1296          PUSHL   #80@16!SS$_NORMAL[                    ; set expected IO status
                1287'CF  01   FB    0FFF  1297          CALLS   #1,W^BUF_CHECK                        ; check the data
  0050 8F    00  01DB'CF  00   2C    1004  1298          MOVC5   #0,W^GETBUF+8,#0,#80,W^GETBUF+8        ; init the buffer
                01DB'CF             100D
                                    1010  1299  ;+
                                    1010  1300  ;
                                    1010  1301  ; test _S with common EFN
                                    1010  1302  ;
                                    1010  1303  ;-
                                    1010  1304          NEXT_TEST
                                    1010
                                    1010        STP41:
                0004'CF  29   D0    1010                          MOVL    #41,W^CURRENT_TC
                         00   DD    1015                          PUSHL   #0
                1385'CF  01   FB    1017                          CALLS   #1,W^REG_SAVE
                                    101C  1305          $QIO_S  CHAN=W^MBCHAN,-
                                    101C  1306                  FUNC=#IO$_WRITEVBLK,-
                                    101C  1307                  P1  =W^TEST_DATA,-
                                    101C  1308                  P2  =#80                              ; set up mailbox
                                    103F  1309          $QIOW_S CHAN=W^MBCHAN,-
                                    103F  1310                  EFN =#65,-
                                    103F  1311                  FUNC=#IO$_READVBLK,-
                                    103F  1312                  P1  =W^GETBUF+8,-
                                    103F  1313                  P2  =#80                              ; try _S with common EFC
                                    1066  1314          FAIL_CHECK SS$_NORMAL                        ; check for success
                         01   DD    1066                          PUSHL   #SS$_NORMAL
                138F'CF  01   FB    1068                          CALLS   #1,W^REG_CHECK
               00500001  8F   DD    106D  1315          PUSHL   #80@16!SS$_NORMAL[                    ; set expected IO status
                1287'CF  01   FB    1073  1316          CALLS   #1,W^BUF_CHECK                        ; check the data
  0050 8F    00  01DB'CF  00   2C    1078  1317          MOVC5   #0,W^GETBUF+8,#0,#80,W^GETBUF+8        ; init the buffer
                01DB'CF             1081
                                    1084  1318  ;+
                                    1084  1319  ;
                                    1084  1320  ; test _G with common EFC
                                    1084  1321  ;
                                    1084  1322  ;-
                                    1084  1323          NEXT_TEST
                                    1084
                                    1084        STP42:
                0004'CF  2A   D0    1084                          MOVL    #42,W^CURRENT_TC
                         00   DD    1089                          PUSHL   #0
                1385'CF  01   FB    108B                          CALLS   #1,W^REG_SAVE
       0129'CF  00000041  8F   D0    1090  1324          MOVL    #65,W^QIOWP+QIOW$_EFN                  ; set EFN
                0131'CF  30   D0    1099  1325          MOVL    #IO$_WRITEVBLK,W^QIOWP+QIOW$_FUNC      ; set function
       0141'CF  0250'CF       DE    109E  1326          MOVAL   W^TEST_DATA,W^QIOWP+QIOW$_P1           ; set new P1 parameter
                                    10A5  1327          $QIO_S  CHAN=W^MBCHAN,-
                                    10A5  1328                  FUNC=#IO$_READVBLK,-
                                    10A5  1329                  P1  =W^GETBUF+8,-
                                    10A5  1330                  P2  =#80                              ; set up mailbox
                                    10C8  1331          $QIOW_G W^QIOWP                               ; try _G with common EFN
                                    10D1  1332          FAIL_CHECK SS$_NORMAL                        ; check for success
                         01   DD    10D1                          PUSHL   #SS$_NORMAL
                138F'CF  01   FB    10D3                          CALLS   #1,W^REG_CHECK
               00500001  8F   DD    10D8  1333          PUSHL   #80@16!SS$_NORMAL[                    ; set expected IO status
```

```
        1287'CF    01   FB  10DE        CALLS   #1,W^BUF_CHECK                  ; check the data
                            10E3  ;+
                            10E3  ;
                            10E3  ; reset super mode handler to the original address and
                            10E3  ; dump any errors on the terminal that occured at AST disable time.
                            10E3  ;
                            10E3  ;-
                            10E3  CLEAN_UP:
                            10E3        $DLCEFC_S W^EFCNAM                      ; get rid of the cluster
                            10EE        $DASSGN_S CHAN=W^MBCHAN                 ; waste the MBXp^/^
   0307'CF   0077'CF    DE  10FA        MOVAL   W^DCLCMH,W^SERV_NAME            ; set service name
                   02   BE  1101        CHMS    #2                              ; reset the CHMS handler
        1AF2'CF    00   FB  1103        CALLS #0,W^ERLBUF_DUMP                  ; dump any errors
                            1108        TEST_END
              004C'CF   DD  1108                PUSHL   W^TMD_ADDR
              0048'CF   DD  110C                PUSHL   W^TMN_ADDR
                   02   DD  1110                PUSHL   #2
              0044'CF   DD  1112                PUSHL   W^MOD_MSG_CODE
     00000000'GF   04   FB  1116                CALLS   #SST1,G^LIB$SIGNAL
0044'CF   01   1C  01   F0  111D                INSV    #1,#STS$V_INHIB_MSG,#1,W^MOD_MSG_CODE
              0044'CF   DD  1124                PUSHL   W^MOD_MSG_CODE
     00000000'GF   01   FB  1128                CALLS   #1,G^SYS$EXIT
```

```
112F  .SBTTL  ROUTINES
112F  .SBTTL SETUP_SUPER ROUTINE
112F  ;++
112F  ;
112F  ;       Routine to declare an initial CHMS handler from user mode.
112F  ;
112F  ; FUNCTIONAL DESCRIPTION:
112F  ;
112F  ; CALLING SEQUENCE:
112F  ;
112F  ;       $CMKRNL_S W^SETUP_SUPER,ARGLST
112F  ;
112F  ;               ARGLST = address of a pointer to a one parameter argument list conta
112F  ;                        the address of the entry mask of the CHMS handler
112F  ;
112F  ; INPUT PARAMETERS:
112F  ;
112F  ;       ARGLST
112F  ;
112F  ; IMPLICIT INPUTS
112F  ;
112F  ;       NONE
112F  ;
112F  ; OUTPUT PARAMETERS:
112F  ;
112F  ;       Declares a change mode handler for super mode which must be
112F  ;       reset to DCL in the users handler routine when the handler is
112F  ;       no longer needed.
112F  ;
112F  ; IMPLICIT OUTPUTS:
112F  ;
112F  ;       NONE
112F  ;
112F  ; COMPLETION CODES:
112F  ;
112F  ;       NONE
112F  ;
112F  ; SIDE EFFECTS:
112F  ;
112F  ;       NONE
112F  ;
112F  ; ON ENTRY:
112F  ;
112F  ;               KSP => !---------!           USP => !---------!
112F  ;                      !    0    !                  !         !
112F  ;                      !    0    !                  !  USER   !
112F  ;                      !   AP    !                  !         !
112F  ;                      !   FP    !                  !  CALL   !
112F  ;                      !   PC    !                  !         !
112F  ;                      !    0    !                  !  FRAME  !
112F  ;                      !    0    !                  !         !
112F  ;                      !   AP    !                  !---------!
112F  ;                      !   FP    !
112F  ;                      !SRVEXIT  !
112F  ;                      !   PC    !
112F  ;                      !   PSL   !
112F  ;                      !---------!
112F  ;
112F  ;--
```

```
                             112F RETURN_PC:
               00000000      112F         .LONG   0                            ; storage for user return PC
                             1133 HANDLER_PC:
               00000000      1133         .LONG   0                            ; storage for handler PC
                             1137 ;
                             1137 SETUP_SUPER:
                   000C      1137         .WORD   ^M<R2,R3>
         53   03     DB      1139         MFPR    #PR$_USP,R3                  ; get the user call frame address
   EE AF   10 A3     D0      113C         MOVL    SF$L_SAVE_PC(R3),B^RETURN_PC ; get the user return PC
   ED AF   04 AC     D0      1141         MOVL    4(AP),HANDLER_PC             ; save the handler address
      52   0C AD     D0      1146         MOVL    SF$L_SAVE_FP(FP),R2          ; get saved FP
         52   00'    C0      114A         ADDL    S^#EXE$C_CMSTKSZ,R2          ; back over change mode stack frame
      62   5B'AF     9E      114D         MOVAB   B^20$,(R2)                   ; set return address
              0A     F0      1151         INSV    #<<PSL$C_SUPER@PSL$S_CURMOD>+PSL$C_SUPER>,-
              16             1153                 #PSL$V_PRVMOD,-
      04 A2   04             1154                 #PSL$S_CURMOD*2,4(R2)        ; set current and previous mode to super
         50   01     D0      1157         MOVL    S^#SS$_NORMAL,R0             ; set correct return code
                     04      115A         RET                                  ; enter super mode
                             115B 20$:
              7E     D4      115B         CLRL    -(SP)                        ; set up dummy PSL
     61'AF    6E     FA      115D         CALLG   (SP),B^30$                   ; create initial call frame
                             1161 30$:
                   0000      1161         .WORD   ^M<>                         ; entry mask
              00     DD      1163         PUSHL   #0                           ; push a dummy parameter
   1385'CF    01     FB      1165         CALLS   #1,W^REG_SAVE                ; save the registers
                             116A         $DCLCMH_S @HANDLER_PC,W^PRVHND1,#0 ; set real handler
                             117A         FAIL_CHECKNP SS$_NORMAL              ; check for success
              01     DD      117A                 PUSHL   #SS$_NORMAL
   1A76'CF    01     FB      117C                 CALLS   #1,W^REG_CHECKNP
  03C00000    8F     DD      1181         PUSHL   #<<PSL$C_USER@PSL$V_CURMOD>-
                             1187                 !<PSL$C_USER@PSL$V_PRVMOD>>; set return to user
           A5 AF     DD      1187         PUSHL   RETURN_PC                    ; set the return PC
                     02      118A         REI                                  ; return to user mode
```

```
                                 11B8  .SBTTL SUPER_MODE
                                 11B8  ;++
                                 11B8  ; FUNCTIONAL DESCRIPTION:
                                 11B8  ;       Routine to handle the CHMS instructions.
                                 11B8  ;
                                 11B8  ; CALLING SEQUENCE:
                                 11B8  ;       CHMS    #N
                                 11B8  ;
                                 11B8  ; INPUT PARAMETERS:
                                 11B8  ;           SP=>  CHMS parameter
                                 11B8  ;                 PC
                                 11B8  ;                 PSL
                                 11B8  ;
                                 11B8  ;           The CHMS parameter can be one of the following:
                                 11B8  ;
                                 11B8  ;                 1 = execute $ASSIGN and $DASSGN service tests
                                 11B8  ;                 2 = execute a $DCLCMH_S to reset the CHMS handler to DCL
                                 11B8  ;                 3 = execute $ALLOC and $DALLOC service tests
                                 11B8  ;
                                 11B8  ; OUTPUT PARAMETERS:
                                 11B8  ;       NONE
                                 11B8  ;--
                                 11B8
                                 11B8  SUPER_MODE:
               50    8E    D0    11B8         MOVL    (SP)+,R0                        ; get CHM parameter off the stack
         03    01    50    8F    11BE         CASEB   R0,#1,#3                        ; do the right thing
                                 1192  10$:
                        0006'    1192         .WORD   20$-10$
                        0010'    1194         .WORD   A30-10$
                        0031'    1196         .WORD   A40-10$
                                 1198  20$:
                     02    DD    1198         PUSHL   #PSL$C_SUPER                    ; push the mode
          1BEF'CF    01    FB    119A         CALLS   #1,W^ASSDAS_CHK                 ; do the tests
                   0028    31    119F         BRW     A50                             ; get back to user mode
                                 11A2  A30:
   0307'CF    0077'CF      DE    11A2         MOVAL   W^DCLCMH,W^SERV_NAME            ; set service name pointer
                                 11A9         $DCLCMH_S @PRVHND1,,#0                  ; reset the CHMS handler for DCL
                                 11BA         FAIL_CHECK SS$_NORMAL                   ; check for success
                     01    DD    11BA                 PUSHL   #SS$_NORMAL
          138F'CF    01    FB    11BC                 CALLS   #1,W^REG_CHECK
                     07    11    11C1         BRB     A50                             ; get back to user mode
                                 11C3  A40:
                     02    DD    11C3         PUSHL   #PSL$C_SUPER                    ; push the mode
          1B5C'CF    01    FB    11C5         CALLS   #1,W^ALLDAL_CHK                 ; do the tests
                                 11CA  A50:
                           02    11CA         REI                                     ; return to user mode
```

```
                                        11CB          .SBTTL BUF_CHECK
                                        11CB    ;++
                                        11CB    ; FUNCTIONAL DESCRIPTION:
                                        11CB    ;       Routine to check the contents of a buffer against known good
                                        11CB    ;       data and check the IO status return.
                                        11CB    ;
                                        11CB    ; CALLING SEQUENCE:
                                        11CB    ;       PUSHL   #EXPECTED_IOSTATUS      ; set expected IO status
                                        11CB    ;       CALLS #1,W^BUF_CHECK            ; check buffer
                                        11CB    ;
                                        11CB    ; INPUT PARAMETERS:
                                        11CB    ;       R6 = buffer address
                                        11CB    ;       R7 = good data address
                                        11CB    ;       R8 = byte count
                                        11CB    ;       STAT = IO status #1
                                        11CB    ;       STAT1 = IO status #2
                                        11CB    ;
                                        11CB    ; OUTPUT PARAMETERS:
                                        11CB    ;       NONE
                                        11CB    ;
                                        11CB    ;--
                                        11CB
                                        11CB    DISK_BUF_CHECK:
                                OFFC    11CB            .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                                        11CD            $GETDVI_S CHAN = CHAN1,-        ; Get characteristics for our disk
                                        11CD                    ITMLST = DISK_ITMLST
         7E    0000037A'EF      B0      11EB            MOVW    PB+8+DIB$W_UNIT,-(SP)   ; Save old device unit number...
 0000037A'EF   00001283'EF      B0      11F2            MOVW    DISK_UNIT,PB+8+DIB$W_UNIT ; ...and substitute our own
               000002EB'EF      DD      11FD            PUSHL   ARGLST1                 ; Save ptr to old device name desc...
 000002EB'EF   00001227'EF      DE      1203            MOVAL   DISK_NAME,ARGLST1       ; ...and substitute our own
                        04 AC   DD      120E            PUSHL   4(AP)
      00001287'EF       01      FB      1211            CALLS   #1,BUF_CHECK            ; Check that we got good data
            000002EB'EF 8ED0            1218            POPL    ARGLST1                 ; Restore old device name desc...
      0000037A'EF       8E      B0      121F            MOVW    (SP)+,PB+8+DIB$W_UNIT   ; ...and unit number
                                04      1226            RET
                                        1227
                                        1227    DISK_ITMLST:                            ; ITMLST for $GETDVI
                                        1227    DISK_NAME:                              ; Note that this becomes desc for name
                           0020 0040    1227            .WORD   64,DVI$_DEVNAM          ; Our disk name
                            00001243'   122B            .ADDRESS DISK_NAME_BUF
                            00001227'   122F            .ADDRESS DISK_NAME              ; Note that we overwrite length!
                           000C 0004    1233            .WORD   4,DVI$_UNIT             ; The unit number of the spindle
                            00001283'   1237            .ADDRESS DISK_UNIT
                            00000000    123B            .LONG   0
                            00000000    123F            .LONG   0                       ; End of $GETDVI ITMLST
                                        1243
                                        1243    DISK_NAME_BUF:                          ; String giving our disk name
                            00001283    1243            .BLKB   64
                                        1283
                                        1283    DISK_UNIT:                              ; Unit number of the spindle
                            00001287    1283            .BLKB   4
                                        1287
                                        1287    BUF_CHECK:
                                03FC    1287            .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9>
                   59     56    D0      1289            MOVL    R6,R9                   ; save a copy of the buffer address
           66      67     58    29      128C            CMPC3   R8,(R7),(R6)            ; check the buffer
                   50     13            1290            BEQL    10$                     ; br if good
 02F3'CF    53    59   C3  1292            SUBL3   R9,R3,W^ARGLST1+8                   ; get buffer offset
 02EF'CF  037A'CF      3C  1298            MOVZWL  W^PB+DIB$W_UNIT+8,W^ARGLST1+4       ; get the unit number
       02F7'CF    61   9A  129F            MOVZBL  (R1),W^ARGLST1+12                   ; get the good data
       02FB'CF    63   9A  12A4            MOVZBL  (R3),W^ARGLST1+16                   ; get the bad data
                           12A9            $GETMSG_S MSGID=#UETP$_DATAER,-
                           12A9                      MSGLEN=W^ML,-
                           12A9                      BUFADR=W^CTRSTR,-
                           12A9                      FLAGS =#1                         ; get the ctrstr
                           12C2            $FAOL_S W^CTRSTR,W^ML,W^GETBUF,W^ARGLST1 ; make it readable
             01CB'CF   DF  12D9            PUSHAL  W^ML                                ; push the desc. address
       13D1'CF    01   FB  12DD            CALLS   #1,W^PRINT_FAIL                     ; print the failure
                           12E2    10$:
   0069'CF    04 AC    D1  12E2            CMPL    4(AP),W^STAT                        ; check status #1
                  06   13  12E8            BEQL    20$                                 ; br if OK
             0069'CF   DD  12EA            PUSHL   W^STAT                              ; else save it
                  0C   11  12EE            BRB     30$                                 ; and continue in common
                           12F0    20$:
   0071'CF    04 AC    D1  12F0            CMPL    4(AP),W^STAT1                       ; check IO status #2
                  10   13  12F6            BEQL    40$                                 ; br if OK
             0071'CF   DD  12F8            PUSHL   W^STAT1                             ; else save it
                           12FC    30$:
               04 AC   DD  12FC            PUSHL   4(AP)                               ; save expected
             0182'CF   DF  12FF            PUSHAL  W^IOEXP                             ; push string variable
       13D1'CF    03   FB  1303            CALLS   #3,W^PRINT_FAIL                     ; print the failure
                           1308    40$:
                  04   04  1308            RET                                         ; return
```

```
                          1309          .SBTTL IONC
                          1309  ;++
                          1309  ; FUNCTIONAL DESCRIPTION:
                          1309  ;       AST routine to service IO AST's for the CANCEL service
                          1309  ;
                          1309  ; CALLING SEQUENCE:
                          1309  ;       Entered via an AST
                          1309  ;
                          1309  ; INPUT PARAMETERS:
                          1309  ;       STAT = CANCEL status return
                          1309  ;
                          1309  ; OUTPUT PARAMETERS:
                          1309  ;       NONE
                          1309  ;
                          1309  ;--
                          1309
                          1309  IONC:
                    03FC  1309          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9>
       1B'AF    00    FB  130B          CALLS   #0,B^CAN_CHECK                  ; check the cancel
                          130F          $WAKE_S                                 ; tell the test to wake up!
                      04  131A          RET                                     ; return
                          131B          .SBTTL CAN_CHECK
                          131B  ;++
                          131B  ; FUNCTIONAL DESCRIPTION:
                          131B  ;       Routine to check the results of a CANCELLED IO.
                          131B  ;
                          131B  ; CALLING SEQUENCE:
                          131B  ;       CALLS #0,W^CAN_CHECK    ; check results
                          131B  ;
                          131B  ; INPUT PARAMETERS:
                          131B  ;       NONE
                          131B  ;
                          131B  ; OUTPUT PARAMETERS:
                          131B  ;       NONE
                          131B  ;
                          131B  ;--
                          131B
                          131B  CAN_CHECK:
                    03FC  131B          .WORD^M<R2,R3,R4,R5,R6,R7,R8,R9>
   2C    0071'CF      B1  131D          CMPW    W^STAT1,#SS$_ABORT              ; check IO status blk
              0F      13  1322          BEQL    10$                             ; br if OK
         0071'CF      DD  1324          PUSHL   W^STAT1                         ; push received
              2C      DD  1328          PUSHL   #SS$_ABORT                      ; push expected
         0174'CF      DF  132A          PUSHAL  W^EXP                           ; push string variable
 13D1'CF      03      FB  132E          CALLS   #3,W^PRINT_FAIL                 ; print the failure
                          1333  10$:
         0069'CF      D4  1333          CLRL    W^STAT                          ; setup for next CANCEL
                      04  1337          RET                                     ; return
```

```
                                 1338          .SBTTL  COUNT_CHAN
                                 1338  ;++
                                 1338  ; FUNCTIONAL DESCRIPTION:
                                 1338  ;       Routine to count the number of assigned channels.
                                 1338  ;
                                 1338  ; CALLING SEQUENCE:
                                 1338  ;       CALLS #0,W^COUNT_CHAN   ; count the number of assigned channels
                                 1338  ;
                                 1338  ; INPUT PARAMETERS:
                                 1338  ;       NONE
                                 1338  ;
                                 1338  ; OUTPUT PARAMETERS:
                                 1338  ;       TOTAL_CHAN = count of all assigned channels
                                 1338  ;
                                 1338  ;--
                                 1338
                                 1338  TOTAL_CHAN:
                       00000000  1338          .LONG   0                               ; assigned channel count
                                 133C  COUNT_CHAN:
                           001C  133C          .WORD   ^M<R2,R3,R4>
   52   09   00000000'EF   C1    133E          ADDL3   CTL$GL_CCBBASE,#CCB$B_AMOD,R2   ; get base and offset to test assign
                   53   10   CE  1346          MNEGL   #CCB$C_LENGTH,R3                ; set starting channel index
        54   00000000'9F   3C    1349          MOVZWL  @#CTL$GW_NMIOCH,R4              ; get number of I/O channels
                FFE4 CF   D4     1350          CLRL    W^TOTAL_CHAN                    ; init the # of channels
                                 1354  10$:
                   6243   95     1354          TSTB    (R2)[R3]                        ; is channel assigned?
                     04   13     1357          BEQL    20$                             ; br if not assigned
                FFDB CF   D6     1359          INCL    W^TOTAL_CHAN                    ; else bump chan count
                                 135D  20$:
                  53   10   C2   135D          SUBL2   #CCB$C_LENGTH,R3                ; calc next channel index
                  F1   54   F5   1360          SOBGTR  R4,10$                          ; any more CCB's?
                           04    1363          RET                                     ; return
                                 1364          .SBTTL STORE_STEP
                                 1364  ;++
                                 1364  ; FUNCTIONAL DESCRIPTION:
                                 1364  ;       Routine to store step information in the error log buffer.
                                 1364  ;
                                 1364  ; CALLING SEQUENCE:
                                 1364  ;       CALLS #0,W^STORE_STEP
                                 1364  ;
                                 1364  ; INPUT PARAMETERS:
                                 1364  ;       ELBP = current errlog buffer pointer
                                 1364  ;
                                 1364  ; OUTPUT PARAMETERS:
                                 1364  ;       FLAG = error logged flag
                                 1364  ;
                                 1364  ;--
                                 1364
                                 1364  STORE_STEP:
                         0004    1364          .WORD   ^M<R2>
          1495'CF    01   88     1366          BISB2   #1,W^FLAG                       ; set the error logged flag
          52   1496'CF   D0      1368          MOVL    W^ELBP,R2                       ; get errlog buf pntr
          82   0307'CF   D0      1370          MOVL    W^SERV_NAME,(R2)+               ; save the service name
          82   0004'CF   D0      1375          MOVL    W^CURRENT_TC,(R2)+              ; save the step number
          82   0159'CF   D0      137A          MOVL    W^MODE,(R2)+                    ; save the mode
          1496'CF    52   D0     137F          MOVL    R2,W^ELBP                       ; reset the errlog buf pntr
                         04      1384          RET                                     ; return
```

```
SA
Sy

TE
TE
TM
TM
TO
TO
TO
TP
UE
UE
UE
UM
WA

PS
--
.
$A
RO
RW
SA

Ph
--
In
Co
Pa
Sy
Pa
Sy
Ps
Cr
As

Th
21
Th
20
10
```

```
                                   1385       .SBTTL REG_SAVE
                                   1385  ;++
                                   1385  ; FUNCTIONAL DESCRIPTION:
                                   1385  ;       Subroutine to save R2-R11 in the register save location.
                                   1385  ;
                                   1385  ; CALLING SEQUENCE:
                                   1385  ;       PUSHL   #0                ; save a dummy parameter
                                   1385  ;       CALLS   #1,W^REG_SAVE     ; save R2-R11
                                   1385  ;
                                   1385  ; INPUT PARAMETERS:
                                   1385  ;       NONE
                                   1385  ;
                                   1385  ; OUTPUT PARAMETERS:
                                   1385  ;       NONE
                                   1385  ;
                                   1385  ;--
                                   1385
                                   1385  REG_SAVE:
                            0FFC   1385         .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
  0008'CF    14 AD    28     28    1387         MOVC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; save the registers in the program
                             04    138E         RET
                                   138F         .SBTTL  REG_CHECK
                                   138F  ;++
                                   138F  ; FUNCTIONAL DESCRIPTION:
                                   138F  ;       Subroutine to test R0 & R2-R11 for proper content after a service
                                   138F  ;       execution. A snapshot is taken by the REG_SAVE routine at the
                                   138F  ;       beginning of each step and this routine is executed after the
                                   138F  ;       services have been executed.
                                   138F  ;
                                   138F  ; CALLING SEQUENCE:
                                   138F  ;       PUSHL   #SS$_XXXXXX       ; push expected R0 contents
                                   138F  ;       CALLS   #1,W^REG_CHECK    ; execute this routine
                                   138F  ;
                                   138F  ; INPUT PARAMETERS:
                                   138F  ;       expected R0 contents on the stack
                                   138F  ;
                                   138F  ; OUTPUT PARAMETERS:
                                   138F  ;       possible error messages printed using $PUTMSG
                                   138F  ;
                                   138F  ;--
                                   138F
                                   138F  REG_CHECK:
                            0FFC   138F         .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
             50    04 AC     D1    1391         CMPL    4(AP),R0                          ; is this the right fail code?
                         0E  13    1395         BEQL    10$                               ; br if yes
                         50  DD    1397         PUSHL   R0                                ; push received data
                      04 AC  DD    1399         PUSHL   4(AP)                             ; push expected data
                    0174'CF  DF    139C         PUSHAL  W^EXP                             ; push the string variable
           13D1'CF    03     FB    13A0         CALLS   #3,W^PRINT_FAIL                   ; print the error message
                                   13A5  10$:
  0008'CF    14 AD    28     29    13A5         CMPC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; check all but R0
                      22     13    13AC         BEQL    20$                               ; br if O.K.
 56    53   00000008'8F      C3    13AE         SUBL3   #REG_SAVE_AREA,R3,R6              ; calculate the register number
                56    04     C6    13B6         DIVL2   #4,R6
          7E    56    02     81    13B9         ADDB3   #^X2,R6,-(SP)                     ; set number past R0-R1 and save
                51    03     CA    13BD         BICL2   #3,R1                             ; backup to register boundrys
                53    03     CA    13C0         BICL2   #3,R3
```

```
                     61  DD  13C3                PUSHL   (R1)                                   ; push received data
                     63  DD  13C5                PUSHL   (R3)                                   ; push expected data
                015D'CF  DF  13C7                PUSHAL  W^REG                                  ; set string pntr param.
        13D1'CF      04  FB  13CB                CALLS   #4,W^PRINT_FAIL                        ; print the error message
                             13D0      20$:
                         04  13D0                RET
                             13D1                .SBTTL  PRINT_FAIL
                             13D1      ;++
                             13D1      ; FUNCTIONAL DESCRIPTION:
                             13D1      ;         Subroutine to report failures using $PUTMSG
                             13D1      ;
                             13D1      ; CALLING SEQUENCE:
                             13D1      ; Mode  #1        PUSHL EXPECTED  Mode    #2        PUSHL REG_NUMBER
                             13D1      ;                 PUSHL RECEIVED                      PUSHL EXPECTED
                             13D1      ;                 PUSHAL STRING_VAR                   PUSHL RECEIVED
                             13D1      ;                 CALLS #3,W^PRINT_FAIL               PUSHAL STRING_VAR
                             13D1      ;                                                     CALLS #4,W^PRINT_FAIL
                             13D1      ; Mode  #3        PUSHAL STRING_VAR
                             13D1      ;                 CALLS #1,W^PRINT_FAIL
                             13D1      ;
                             13D1      ; INPUT PARAMETERS:
                             13D1      ;         listed above
                             13D1      ;
                             13D1      ; OUTPUT PARAMETERS:
                             13D1      ;         an error message is printed using $PUTMSG
                             13D1      ;
                             13D1      ;--
                             13D1
                             13D1      PRINT_FAIL:
                       003C  13D1                .WORD   ^M<R2,R3,R4,R5>
                             13D3                $FAO_S  W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
                             13F4                $PUTMSG_S W^MSGVEC                             ; print the message
              04     6C  91  1405                CMPB    (AP),#4                                ; is this a register message?
                     26  13  1408                BEQL    10$                                    ; br if yes
              01     6C  91  140A                CMPB    (AP),#1                                ; is this just a message?
                     48  13  140D                BEQL    20$                                    ; br if yes
                             140F                $FAO_S  W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
                     40  11  142E                BRB     30$                                    ; goto output message
                             1430      10$:
                             1430                $FAO_S  W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
                     19  11  1455                BRB     30$                                    ; goto output message
                             1457      20$:
    0332'CF      04  AC  D0  1457                MOVL    4(AP),W^MSGVEC1+12                     ; save string address
                             145D                $PUTMSG_S W^MSGVEC1                            ; print the message
                     11  11  146E                BRB     40$                                    ; skip the other message
                             1470      30$:
                             1470                $PUTMSG_S W^MSGVEC                             ; print the message
                             1481      40$:
        1B2F'CF      00  FB  1481                CALLS   #0,W^MODE_ID                           ; identify the mode
    004C'CF     002A'CF  DE  1486                MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR             ; set failure message address
0044'CF    03    00  02  F0  148D                INSV    #ERROR,#0,#3,W^MOD_MSG_CODE            ; set severity code
                         04  1494                RET
```

```
                              1495        .SBTTL REG_CHECKNP
                              1495  ;++
                              1495  ; FUNCTIONAL DESCRIPTION:
                              1495  ;       Subroutine to test R0 & R2-R11 for proper content after a service
                              1495  ;       execution without printing it. A snapshot is taken by the REG_SAVE routine a
                              1495  ;       beginning of each step and this routine is executed after the
                              1495  ;       services have been executed. This routine collects the error
                              1495  ;       information in buffer ERLB instead of printing it.
                              1495  ;
                              1495  ; CALLING SEQUENCE:
                              1495  ;       PUSHL   #SS$_XXXXXX     ; push expected R0 contents
                              1495  ;       CALLS   #1,W^REG_CHECK  ; execute this routine
                              1495  ;
                              1495  ; INPUT PARAMETERS:
                              1495  ;       expected R0 contents on the stack
                              1495  ;
                              1495  ; OUTPUT PARAMETERS:
                              1495  ;       possible error messages logged in buffer ERLB which are printed
                              1495  ;       using routine ERLBUF_DUMP.
                              1495  ;
                              1495  ;       Error packets are in the following form:
                              1495  ;
                              1495  ;               !-----------------!
                              1495  ;               !Service name pntr!
                              1495  ;               !-----------------!
                              1495  ;               !     Step #      !
                              1495  ;               !-----------------!
                              1495  ;               !Mode name pointer!
                              1495  ;               !-----------------!
                              1495  ;               !               ! ! long word count
                              1495  ;               !-----------------!
                              1495  ;               !\/\/\/\/\/\/\/\/\! 3-4 parameter long words
                              1495  ;
                              1495  ;--
                              1495
                              1495  FLAG:
                        00    1495        .BYTE 0                 ; error flags are BIT0 = 0 means no errors in the bu
                              1496                                ;                 BIT0 = 1 means errors in the buffe
                              1496  ELBP:
                 0000149A'    1496        .ADDRESS ERLB           ; error log buffer pointer
                              149A  ERLB:
                 00001A76     149A        .BLKB   1500            ; error log buffer
                              1A76  ;
                              1A76  REG_CHECKNP:
                      0FFC    1A76        .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
      50    04 AC       D1    1A78        CMPL    4(AP),R0        ; is this the right fail code
                 2E     13    1A7C        BEQL    10$             ; br if yes
    F8E1 CF    00       FB    1A7E        CALLS #0,W^STORE_STEP   ; store step information
    52   FA0F  CF       D0    1A83        MOVL    ELBP,R2         ; get the current error log pointer
          82   03       90    1A88        MOVB    #3,(R2)+        ; save the long word count
          82   50       D0    1A8B        MOVL    R0,(R2)+        ; save received status
      82    04 AC       D0    1A8E        MOVL    4(AP),(R2)+     ; save expected status
    82   0174'CF        DE    1A92        MOVAL   W^EXP,(R2)+     ; save the string variable
                 62     D4    1A97        CLRL    (R2)            ; set the terminator
    F9F8 CF    52       D0    1A99        MOVL    R2,ELBP         ; reset the buffer pointer
  004C'CF   002A'CF     DE    1A9E        MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR ; set failure message address
0044'CF   03   00   02  F0    1AA5        INSV    #ERROR,#0,#3,W^MOD_MSG_CODE ; set severity code
```

```
                                 1AAC  10$:
  0008'CF    14 AD    28    29   1AAC         CMPC3   #4*10,^X14(FP),W^REG_SAVE_AREA ; check all but R0 and R1
                      3C    13   1AB3         BEQL    20$                 ; br if OK
          F8AA CF     00    FB   1AB5         CALLS #0,W^STORE_STEP       ; store step information
          52   F9D8   CF    D0   1ABA         MOVL    ELBP,R2             ; get current error log buf pointer
                 82   04    90   1ABF         MOVB    S^#4,(R2)+          ; set longword count
            00000008'8F     C3   1AC2         SUBL3   #REG_SAVE_AREA,-
                 56   53         1AC8         R3,R6                       ; calc reg number
                 56   04    C6   1ACA         DIVL2   S^#4,R6             ; make it a longword count
          82     56   02    C1   1ACD         ADDL3   S^#2,R6,(R2)+       ; correct for R0-R1 and save
                 82   61    D0   1AD1         MOVL    (R1),(R2)+          ; save received results
                 82   63    D0   1AD4         MOVL    (R3),(R2)+          ; save expected results
          82   015D'CF      DE   1AD7         MOVAL   W^REG,(R2)+         ; save string variable
                      62    D4   1ADC         CLRL    (R2)                ; set the terminator
          F9B3 CF     52    D0   1ADE         MOVL    R2,ELBP             ; reset the buffer pointer
     004C'CF   002A'CF      DE   1AE3         MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR       ; set failure message address
  0044'CF    03   00   02   F0   1AEA         INSV    #ERROR,#0,#3,W^MOD_MSG_CODE      ; set severity code
                                 1AF1  20$:
                            04   1AF1         RET                                ; bail out
```

```
                          1AF2          .SBTTL  ERLBUF_DUMP
                          1AF2  ;++
                          1AF2  ; FUNCTIONAL DESCRIPTION:
                          1AF2  ;       Routine to check for errors in the error log buffer and
                          1AF2  ;       report any that are there.
                          1AF2  ;
                          1AF2  ; CALLING SEQUENCE:
                          1AF2  ;       CALLS #0,W^ERLBUF_DUMP
                          1AF2  ;
                          1AF2  ; INPUT PARAMETERS:
                          1AF2  ;       FLAG bit 0 = 0 for no errors logged
                          1AF2  ;       FLAG bit 0 = 1 for errors logged
                          1AF2  ;       if errors logged then buffer ERLB must contain legal format errors
                          1AF2  ;
                          1AF2  ; OUTPUT PARAMETERS:
                          1AF2  ;       NONE
                          1AF2  ;
                          1AF2  ;--
                          1AF2
                          1AF2  ERLBUF_DUMP:
                    001C  1AF2          .WORD   ^M<R2,R3,R4>
      2A F99D CF     E9   1AF4          BLBC    FLAG,30$            ; br if no errors to report
    52   F99D CF     DE   1AF9          MOVAL   ERLB,R2             ; set up buffer pointer
                          1AFE  10$:
                62   D5   1AFE          TSTL    (R2)                ; any more errors?
                21   13   1B00          BEQL    30$                 ; br if not
     0307'CF    82   D0   1B02          MOVL    (R2)+,W^SERV_NAME   ; reset service name
     0004'CF    82   D0   1B07          MOVL    (R2)+,W^CURRENT_TC  ; reset step #
     0159'CF    82   D0   1B0C          MOVL    (R2)+,W^MODE        ; reset the mode
          53    82   9A   1B11          MOVZBL  (R2)+,R3            ; get the longword count
          54    53   D0   1B14          MOVL    R3,R4               ; and save it
                          1B17  20$:
                82   DD   1B17          PUSHL   (R2)+               ; push a parameter
          FB    53   F5   1B19          SOBGTR  R3,20$              ; and push them all
     F880 CF    54   FB   1B1C          CALLS   R4,W^PRINT_FAIL     ; print the failure
                DB   11   1B21          BRB     10$                 ; do the next one
                          1B23  30$:
F96C CF   F973 CF    DE   1B23          MOVAL   W^ERLB,W^ELBP       ; reset the buffer pointer
          F96C CF    D4   1B2A          CLRL    W^ERLB              ; set fresh terminater
                     04   1B2E          RET                         ; bail out
```

```
                                1B2F  1905          .SBTTL  MODE_ID
                                1B2F  1906  ;++
                                1B2F  1907  ; FUNCTIONAL DESCRIPTION:
                                1B2F  1908  ;       Subroutine to identify the mode that an exit handler is in.
                                1B2F  1909  ;
                                1B2F  1910  ; CALLING SEQUENCE:
                                1B2F  1911  ;       CALLS   #0,W^MODE_ID
                                1B2F  1912  ;
                                1B2F  1913  ; INPUT PARAMETERS:
                                1B2F  1914  ;       MODE contains an address pointing to an ascii string desc.
                                1B2F  1915  ;       of the current CPU mode.
                                1B2F  1916  ;
                                1B2F  1917  ; OUTPUT PARAMETERS:
                                1B2F  1918  ;       NONE
                                1B2F  1919  ;
                                1B2F  1920  ;--
                                1B2F  1921
                                1B2F  1922  MODE_ID:
                          003C  1B2F  1923          .WORD   ^M<R2,R3,R4,R5>
                                1B31  1924          $FAO_S  W^CS5,W^MESSAGEL,W^MSGL,MODE ; format the error message
                                1B4A  1925          $PUTMSG_S W^MSGVEC              ; print the mode message
                            04  1B5B  1926          RET
                                1B5C  1927  ;
                                1B5C  1928          .SBTTL  ALLDAL_CHK
                                1B5C  1929  ;++
                                1B5C  1930  ; FUNCTIONAL DESCRIPTION:
                                1B5C  1931  ;       Subroutine to do the $ALLOC and $DALLOC tests
                                1B5C  1932  ;
                                1B5C  1933  ; CALLING SEQUENCE:
                                1B5C  1934  ;       PUSHL   #ACCESS_MODE
                                1B5C  1935  ;       CALLS   #1,W^ALLDAL_CHK
                                1B5C  1936  ;
                                1B5C  1937  ; INPUT PARAMETERS:
                                1B5C  1938  ;       4(AP) = the access mode for the test
                                1B5C  1939  ;
                                1B5C  1940  ; OUTPUT PARAMETERS:
                                1B5C  1941  ;       NONE
                                1B5C  1942  ;
                                1B5C  1943  ;--
                                1B5C  1944
                                1B5C  1945  ALLDAL_CHK:
                          003C  1B5C  1946          .WORD   ^M<R2,R3,R4,R5>
                      00    DD  1B5E  1947          PUSHL   #0                      ; push a dummy parameter
         F820 CF      01    FB  1B60  1948          CALLS   #1,W^REG_SAVE           ; save a register snapshot
                                1B65  1949          $ALLOC_S DEVNAM=@^MBNAM,-
                                1B65  1950                  PHYLEN=W^ML,-
                                1B65  1951                  PHYBUF=W^GETBUF,-
                                1B65  1952                  ACMODE=4(AP)            ; try _S mode
                                1B7D  1953          FAIL_CHECKNP SS$_NORMAL         ; check for success
                      01    DD  1B7D                    PUSHL   #SS$_NORMAL
         FEF2 CF      01    FB  1B7F                    CALLS   #1,W^REG_CHECKNP
      009D'CF      04 AC    D0  1B84  1954          MOVL    4(AP),W^ALLO+ALLOC$_ACMODE ; set the new access mode
                                1B8A  1955          $ALLOC_G W^ALLO                 ; try _G mode
                                1B93  1956          FAIL_CHECKNP SS$_DEVALRALLOC    ; check for proper failure
         00000641 8F        DD  1B93                    PUSHL   #SS$_DEVALRALLOC
         FED8 CF      01    FB  1B99                    CALLS   #1,W^REG_CHECKNP
   0307'CF    004C'CF       DE  1B9E  1957          MOVAL   W^DALLOC,W^SERV_NAME    ; set new service name
```

```
                              1BA5        $DALLOC_S DEVNAM=W^MBNAM,-
                              1BA5                ACMODE=4(AP)            ; try _S mode
                              1BB3        FAIL_CHECKNP SS$_NORMAL         ; check for success
                 01      DD   1BB3                PUSHL   #SS$_NORMAL
        FEBC CF  01      FB   1BB5                CALLS   #1,W^REG_CHECKNP
   0307'CF   0038'CF     DE   1BBA        MOVAL W^ALLOC,W^SERV_NAME       ; set new service name
                              1BC1        $ALLOC_G W^ALLO                 ; try successful _G form
                              1BCA        FAIL_CHECKNP SS$_NORMAL         ; check for success
                 01      DD   1BCA                PUSHL   #SS$_NORMAL
        FEA5 CF  01      FB   1BCC                CALLS   #1,W^REG_CHECKNP
   0307'CF   004C'CF     DE   1BD1        MOVAL   W^DALLOC,W^SERV_NAME    ; set new service name
     00BD'CF   04 AC     D0   1BD8        MOVL    4(AP),W^DALL+DALLOC$_ACMODE ; set new access mode
                              1BDE        $DALLOC_G W^DALL                ; try _G mode
                              1BE7        FAIL_CHECKNP SS$_NORMAL         ; check for success
                 01      DD   1BE7                PUSHL   #SS$_NORMAL
        FE88 CF  01      FB   1BE9                CALLS   #1,W^REG_CHECKNP
                 04           1BEE        RET                             ; return
                              1BEF  ;
                              1BEF        .SBTTL  ASSDAS_CHK
                              1BEF  ;++
                              1BEF  ; FUNCTIONAL DESCRIPTION:
                              1BEF        Subroutine to do the $ASSIGN and $DASSGN tests
                              1BEF  ;
                              1BEF  ; CALLING SEQUENCE:
                              1BEF        PUSHL   #ACCESS_MODE
                              1BEF        CALLS   #1,W^ASSDAS_CHK
                              1BEF  ;
                              1BEF  ; INPUT PARAMETERS:
                              1BEF        4(AP) = the access mode for the test
                              1BEF        CHAN_SAVE = correct number of channels
                              1BEF  ;
                              1BEF  ; OUTPUT PARAMETERS:
                              1BEF        NONE
                              1BEF  ;
                              1BEF  ;--
                              1BEF
                              1BEF  ASSDAS_CHK:
               003C           1BEF        .WORD   ^M<R2,R3,R4,R5>
                 00      DD   1BF1        PUSHL   #0                      ; push a dummy parameter
        F78D CF  01      FB   1BF3        CALLS   #1,W^REG_SAVE           ; save a register snapshot
                              1BF8        $CREMBX_S CHAN=W^MBCHAN,-
                              1BF8                LOGNAM=W^MBNAM,-
                              1BF8                PRMFLG=#0,-
                              1BF8                ACMODE=#PSL$C_USER      ; create temp mailbox
                              1C0F        $ASSIGN_S DEVNAM=W^MBNAM,-
                              1C0F                CHAN  =W^CHAN1,-
                              1C0F                ACMODE=4(AP)            ; try _S mode
                              1C23        FAIL_CHECKNP SS$_NORMAL         ; check success
                 01      DD   1C23                PUSHL   #SS$_NORMAL
        FE4C CF  01      FB   1C25                CALLS   #1,W^REG_CHECKNP
     0085'CF   04 AC     D0   1C2A        MOVL    4(AP),W^ASGN+ASSIGN$_ACMODE ; set the new mode
                              1C30        $ASSIGN_G W^ASGN                ; try the _G form
                              1C39        FAIL_CHECKNP SS$_NORMAL         ; check success
                 01      DD   1C39                PUSHL   #SS$_NORMAL
        FE36 CF  01      FB   1C3B                CALLS   #1,W^REG_CHECKNP
   0307'CF   0045'CF     DE   1C40        MOVAL   W^DASSGN,W^SERV_NAME    ; set service name
                              1C47        $DASSGN_S CHAN=W^CHAN1          ; release channel
```

```
                                1C53   2005           FAIL_CHECKNP SS$_NORMAL            ; check success
                        01  DD  1C53                          PUSHL   #SS$_NORMAL
             FE1C CF    01  FB  1C55                          CALLS   #1,W^REG_CHECKNP
       00B1'CF   0322'CF    D0  1C5A   2006           MOVL    W^CHAN2,W^DASS+DASSGN$_CHAN ; set channel number
                                1C61   2007           $DASSGN_G W^DASS                  ; try _G form
                                1C6A   2008           FAIL_CHECKNP SS$_NORMAL            ; check success
                        01  DD  1C6A                          PUSHL   #SS$_NORMAL
             FE05 CF    01  FB  1C6C                          CALLS   #1,W^REG_CHECKNP
                                1C71   2009           $DASSGN_S CHAN=W^MBCHAN           ; get rid of the mailbox
                                1C7D   2010           FAIL_CHECKNP SS$_NORMAL            ; check success
                        01  DD  1C7D                          PUSHL   #SS$_NORMAL
             FDF2 CF    01  FB  1C7F                          CALLS   #1,W^REG_CHECKNP
                   0320'CF  B5  1C84   2011           TSTW    W^CHAN1                   ; is there a channel #1
                        06  13  1C88   2012           BEQL    10$                       ; br if error
                   0322'CF  B5  1C8A   2013           TSTW    W^CHAN2                   ; is there a channel #2
                        20  12  1C8E   2014           BNEQ    20$                       ; br if no error
                                1C90   2015 10$:
       0307'CF   0031'CF    DE  1C90   2016           MOVAL   W^ASSIGN,W^SERV_NAME      ; set service name
            F6C8 CF     00  FB  1C97   2017           CALLS   #0,W^STORE_STEP           ; save the step information
            52     F7F6 CF  D0  1C9C   2018           MOVL    W^ELBP,R2                 ; get error log buf pntr
                   82   01  90  1CA1   2019           MOVB    #1,(R2)+                  ; save longword count
            82    0139'CF   DE  1CA4   2020           MOVAL   W^CS4,(R2)+               ; save string variable
                        62  D4  1CA9   2021           CLRL    (R2)                      ; set new terminator
            F7E6 CF     52  D0  1CAB   2022           MOVL    R2,W^ELBP                 ; reset the buffer pointer
                                1CB0   2023 20$:
            F687 CF     00  FB  1CB0   2024           CALLS   #0,W^COUNT_CHAN           ; check the number of assigned channels
       0324'CF     F67F CF  D1  1CB5   2025           CMPL    W^TOTAL_CHAN,W^CHAN_SAVE ; correct # of channels?
                        2A  13  1CBC   2026           BEQL    30$                       ; br if OK
       0307'CF   0045'CF    DE  1CBE   2027           MOVAL   W^DASSGN,W^SERV_NAME      ; set service name
            F69A CF     00  FB  1CC5   2028           CALLS   #0,W^STORE_STEP           ; save the step information
            52     F7C8 CF  D0  1CCA   2029           MOVL    W^ELBP,R2                 ; get error log buf pointer
                   82   03  90  1CCF   2030           MOVB    #3,(R2)+                  ; save long word count
            82     F662 CF  3C  1CD2   2031           MOVZWL  W^TOTAL_CHAN,(R2)+        ; save the received count
            82    0324'CF   D0  1CD7   2032           MOVL    W^CHAN_SAVE,(R2)+         ; save expected count
            82    01B8'CF   DE  1CDC   2033           MOVAL   W^IOCC,(R2)+              ; save string variable
                        62  D4  1CE1   2034           CLRL    (R2)                      ; set a new terminator
            F7AE CF     52  D0  1CE3   2035           MOVL    R2,W^ELBP                 ; reset buffer pointer
                                1CE8   2036 30$:
                            04  1CE8   2037           RET                               ; return
```

```
                         1CE9  MOD_MSG_PRINT:
                         1CE9  ;
                         1CE9  ;   ***************************************************************
                         1CE9  ;   *                                                             *
                         1CE9  ;   *  PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES    *
                         1CE9  ;   *       (USING THE PUTMSG MACRO).                             *
                         1CE9  ;   *                                                             *
                         1CE9  ;   ***************************************************************
                         1CE9  ;
                         1CE9          PUTMSG  <MOD_MSG_CODE,#2,TMN_ADDR,TMD_ADDR> ; PRINT MSG
                     05  1D04          RSB                                 ; ... AND RETURN TO CALLER
                         1D05  ;
                         1D05  CHMRTN:
                         1D05  ; *****************************************************************
                         1D05  ; *                                                               *
                         1D05  ; *     CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER   *
                         1D05  ; *     A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED       *
                         1D05  ; *     BY THE MODE MACRO ('TO' OPTION).  IT MERELY DOES          *
                         1D05  ; *     A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS         *
                         1D05  ; *     THE EFFECT OF RETURNING TO THE END OF THE MODE            *
                         1D05  ; *     MACRO EXPANSION.                                          *
                         1D05  ; *                                                               *
                         1D05  ; *****************************************************************
                         1D05  ;
                   0000  1D05          .WORD   0                           ; ENTRY MASK
       00000059'FF   17  1D07          JMP     @CHM_CONT                   ; RETURN TO MODE MACRO IN NEW MODE
                         1D0D  ;
                         1D0D  ; *     RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
                         1D0D  ;
                         1D0D          .END    SATSSS01
```

```
                                                        C  7
SATSSS01                    - SATS SYSTEM SERVICE TESTS  (SUCC S.C.) 16-SEP-1984 00:44:47  VAX/VMS Macro V04-00        Page  54
Symbol table                                                           5-SEP-1984 04:29:37  [UETPSY.SRC]SATSSS01.MAR;1         (6)

$$ARGS              = 0000000C              CTL$GW_NMIOCH         ********     X   04
$$T1                = 00000004              CTRSTR                0000025F R       03
$$T2                = 00000004              CURRENT_TC            00000004 R       03
A                   = 00000084              DALL                  000000B5 R       03
A30                   000011A2 R       04   DALLOC                0000004C R       02
A40                   000011C3 R       04   DALLOC$_ACMODE      = 00000008
A50                   000011CA R       04   DALLOC$_DEVNAM      = 00000004
ALLDAL_CHK            00001B5C R       04   DALLOC$_NARGS       = 00000002
ALLO                  0000008D R       03   DASS                  000000AD R       03
ALLOC                 00000038 R       02   DASSGN                00000045 R       02
ALLOC$_ACMODE       = 00000010              DASSGN$_CHAN        = 00000004
ALLOC$_DEVNAM       = 00000004              DASSGN$_NARGS       = 00000001
ALLOC$_FLAGS        = 00000014              DC$_MAILBOX         = 000000A0
ALLOC$_NARGS        = 00000005              DCLCMH                00000077 R       02
ALLOC$_PHYBUF       = 0000000C              DEV$M_AVL           = 00040000
ALLOC$_PHYLEN       = 00000008              DEV$M_IDV           = 04000000
ARGLST                000002D4 R       02   DEV$M_MBX           = 00100000
ARGLST1               000002EB R       03   DEV$M_ODV           = 08000000
ASGN                  00000079 R       03   DEV$M_REC           = 00000001
ASSDAS_CHK            00001BEF R       04   DEV$M_SHR           = 00010000
ASSIGN                00000031 R       02   DIB$K_LENGTH        = 00000074
ASSIGN$_ACMODE      = 0000000C              DIB$W_UNIT          = 0000000C
ASSIGN$_CHAN        = 00000008              DISALC                000001A5 R       02
ASSIGN$_DEVNAM      = 00000004              DISK                  00000097 R       02
ASSIGN$_MBXNAM      = 00000010              DISK_BUF_CHECK        000011CB R       04
ASSIGN$_NARGS       = 00000004              DISK_ITMLST           00001227 R       04
AST1                  000008CD R       04   DISK_NAME             00001227 R       04
AST2                  000008F0 R       04   DISK_NAME_BUF         00001243 R       04
AST3                  0000097F R       04   DISK_UNIT             00001283 R       04
AST4                  00000A3A R       04   DOT_DIR_SEMI          000004C4 R       03
ASTEXP                00000193 R       02   DOT_DIR_SEMI_LENGTH = 00000006
ATR                   0000048F R       03   DT$_MBX             = 00000001
ATR$C_ASCNAME       = 00000010              DVI$_DEVNAM         = 00000020
ATR$S_ASCNAME       = 00000056              DVI$_UNIT           = 0000000C
BUF                   0000017B R       03   EFCNAM                00000241 R       02
BUF_CHECK             00001287 R       04   ELBP                  00001496 R       04
CANC                  000000A5 R       03   EM                    00000217 R       02
CANCEL                0000003E R       02   ERLB                  0000149A R       04
CANCEL$_CHAN        = 00000004              ERLBUF_DUMP           00001AF2 R       04
CANCEL$_NARGS       = 00000001              ERROR               = 00000002
CAN_CHECK             0000131B R       04   EXE$C_CMSTKSZ         ********     X   04
CCB$B_AMOD          = 00000009              EXP                   00000174 R       02
CCB$C_LENGTH        = 00000010              FIB                   00000466 R       03
CHAN1                 00000320 R       03   FIB$L_ACCTL         = 00000000
CHAN2                 00000322 R       03   FIB$L_EXSZ          = 00000018
CHAN_SAVE             00000324 R       03   FIB$L_EXVBN         = 0000001C
CHMRTN                00001D05 R       04   FIB$L_LOC_ADDR      = 00000028
CHM_CONT              00000059 R       03   FIB$L_WCC           = 00000010
CLEAN_UP              000010E3 R       04   FIB$M_ALCON         = 00000001
COUNT_CHAN            0000133C R       04   FIB$M_EXTEND        = 00000080
CS1                   000000A7 R       02   FIB$M_FILCON        = 00000004
CS2                   000000D9 R       02   FIB$M_NOREAD        = 00000400
CS3                   00000106 R       02   FIB$M_NOWRITE       = 00000001
CS4                   00000139 R       02   FIB$M_SUPERSEDE     = 00000400
CS5                   0000015F R       02   FIB$M_WRITE         = 00000100
CTL$GL_CCBBASE        ********     X   04   FIB$W_DID           = 0000000A
CTL$GL_PHD            ********     X   04   FIB$W_EXCTL         = 00000016
```

```
                                                  D 7
SATSSS01                    - SATS SYSTEM SERVICE TESTS  (SUCC S.C.) 16-SEP-1984 00:44:47  VAX/VMS Macro V04-00       Page  55
Symbol table                                                          5-SEP-1984 04:29:37  [UETPSY.SRC]SATSSS01.MAR;1         (6)

FIB$W_FID                       = 00000004                   MFD_FILE_ID                     = 00040004
FIB$W_FID_RVN                   = 00000008                   ML                                00000108 R     03
FIB$W_NMCTL                     = 00000014                   MODE                              00000159 R     03
FIBDES                            0000045E R     03          MODE_ID                           0000182F R     04
FIBSIZE                         = 00000029                   MOD_MSG_CODE                      00000044 R     03
FILENAME                          0000049B R     03          MOD_MSG_PRINT                     00001CE9 R     04
FILNOTMOD                         000001CB R     02          MSGC                              00000173 R     03
FLAG                              00001495 R     04          MSGVEC                            000002DC R     02
GETBUF                            000001D3 R     03          MSGVEC1                           00000326 R     03
GETC                              000000C1 R     03          NEXT                              00000913 R     04
GETCHN                            00000059 R     02          NEXT1                             000009C4 R     04
GETCHN$_CHAN                    = 00000004                   NEXT2                             00000A86 R     04
GETCHN$_NARGS                   = 00000005                   OUTPUT                            00000067 R     02
GETCHN$_PRIBUF                  = 0000000C                   PB                                00000366 R     03
GETCHN$_PRILEN                  = 00000008                   PHD$Q_PRIVMSK                   = 00000000
GETCHN$_SCDBUF                  = 00000014                   PL                                0000035E R     03
GETCHN$_SCDLEN                  = 00000010                   PR$_USP                         = 00000003
GETD                              000000D9 R     03          PRINT_FAIL                        000013D1 R     04
GETDEV                            00000060 R     02          PRIVMASK                          00000051 R     03
GETDEV$_DEVNAM                  = 00000004                   PRIV_ARGS                       = 00000002
GETDEV$_NARGS                   = 00000005                   PRV$V_SYSPRV                    = 0000001C
GETDEV$_PRIBUF                  = 0000000C                   PRVHND1                           0000030B R     03
GETDEV$_PRILEN                  = 00000008                   PRVPRT                            00000050 R     03
GETDEV$_SCDBUF                  = 00000014                   PSL$C_EXEC                      = 00000001
GETDEV$_SCDLEN                  = 00000010                   PSL$C_KERNEL                    = 00000000
HANDLER_PC                        00001133 R     04          PSL$C_SUPER                     = 00000002
INFO                            = 00000003                   PSL$C_USER                      = 00000003
INPUT                             00000053 R     02          PSL$S_CURMOD                    = 00000002
IO$M_ACCESS                     = 00000040                   PSL$V_CURMOD                    = 00000018
IO$M_CREATE                     = 00000080                   PSL$V_PRVMOD                    = 00000016
IO$M_DELETE                     = 00000100                   QIO                               0000006E R     02
IO$M_READATTN                   = 00000080                   QIO$_ASTADR                     = 00000014
IO$M_WRTATTN                    = 00000100                   QIO$_ASTPRM                     = 00000018
IO$_ACCESS                      = 00000032                   QIO$_CHAN                       = 00000008
IO$_CREATE                      = 00000033                   QIO$_EFN                        = 00000004
IO$_DEACCESS                    = 00000034                   QIO$_FUNC                       = 0000000C
IO$_DELETE                      = 00000035                   QIO$_IOSB                       = 00000010
IO$_MODIFY                      = 00000036                   QIO$_NARGS                      = 0000000C
IO$_READLBLK                    = 00000021                   QIO$_P1                         = 0000001C
IO$_READPBLK                    = 0000000C                   QIO$_P2                         = 00000020
IO$_READVBLK                    = 00000031                   QIO$_P3                         = 00000024
IO$_SETMODE                     = 00000023                   QIO$_P4                         = 00000028
IO$_WRITELBLK                   = 00000020                   QIO$_P5                         = 0000002C
IO$_WRITEOF                     = 00000028                   QIO$_P6                         = 00000030
IO$_WRITEPBLK                   = 00000008                   QIOP                              000000F1 R     03
IO$_WRITEVBLK                   = 00000030                   QIOW                              00000072 R     02
IOCC                              000001B8 R     02          QIOW$_ASTADR                    = 00000014
IOEXP                             00000182 R     02          QIOW$_ASTPRM                    = 00000018
IONC                              00001309 R     04          QIOW$_CHAN                      = 00000008
KM                                00000228 R     02          QIOW$_EFN                       = 00000004
LIB$SIGNAL                        ********     X 04          QIOW$_FUNC                      = 0000000C
MBA                               00000236 R     02          QIOW$_IOSB                      = 00000010
MBCHAN                            0000031E R     03          QIOW$_NARGS                     = 0000000C
MBNAM                             0000030F R     03          QIOW$_P1                        = 0000001C
MB_CHAR_SIZE                    = 00000028                   QIOW$_P2                        = 00000020
MB_DEV_CHAR                       00000336 R     03          QIOW$_P3                        = 00000024
MESSAGEL                          000002FF R     03          QIOW$_P4                        = 00000028
```

```
                                                  E  7
SATSSS01                    - SATS SYSTEM SERVICE TESTS  (SUCC S.C.) 16-SEP-1984 00:44:47  VAX/VMS Macro V04-00        Page  56
Symbol table                                                          5-SEP-1984 04:29:37  [UETPSY.SRC]SATSSS01.MAR;1       (6)

QIOW$_P5                       = 0000002C                  STP3                         00000108 R      04
QIOW$_P6                       = 00000030                  STP30                        000009C4 R      04
QIOWP                            00000125 R      03        STP31                        00000A3C R      04
REG                              0000015D R      03        STP32                        00000A86 R      04
REGNUM                           0000016F R      03        STP33                        00000B0B R      04
REG_CHECK                        0000138F R      04        STP34                        00000C1A R      04
REG_CHECKNP                      00001A76 R      04        STP35                        00000CFE R      04
REG_SAVE                         00001385 R      04        STP36                        00000DA7 R      04
REG_SAVE_AREA                    00000008 R      03        STP37                        00000E45 R      04
RENAST                           0000007E R      02        STP38                        00000EB7 R      04
RETADR                           0000005D R      03        STP39                        00000F20 R      04
RETURN_PC                        0000112F R      04        STP4                         0000014E R      04
SATSSS01                         00000000 RG     04        STP40                        00000FB3 R      04
SB                               000003E2 R      03        STP41                        00001010 R      04
SERV_NAME                        00000307 R      03        STP42                        00001084 R      04
SETUP_SUPER                      00001137 R      04        STP5                         00000197 R      04
SEVERE                         = 00000004                  STP6                         000001B3 R      04
SF$L_SAVE_FP                   = 0000000C                  STP7                         000001E5 R      04
SF$L_SAVE_PC                   = 00000010                  STP8                         00000240 R      04
SHR$K_SHRDEF                   = 00000001                  STP9                         000002A9 R      04
SHR$_TEXT                      = 00001130                  STS$V_INHIB_MSG            = 0000001C
SL                               00000362 R      03        SUCCESS                    = 00000001
SM                               0000020A R      02        SUPER_MODE                   0000118B R      04
SS$_ABORT                      = 0000002C                  SYS$ALLOC                    ********  GX    04
SS$_DEVALRALLOC                = 00000641                  SYS$ASCEFC                   ********  GX    04
SS$_ENDOFFILE                  = 00000870                  SYS$ASSIGN                   ********  GX    04
SS$_NORMAL                     = 00000001                  SYS$CANCEL                   ********  GX    04
SS$_NOSUCHDEV                  = 00000908                  SYS$CMEXEC                   ********  GX    04
SS$_NOTRAN                     = 00000629                  SYS$CMKRNL                   ********  GX    04
STAT                             00000069 R      03        SYS$CREMBX                   ********  GX    04
STAT1                            00000071 R      03        SYS$DALLOC                   ********  GX    04
STATUS                           00000065 R      03        SYS$DASSGN                   ********  GX    04
STEP                           = 0000002A                  SYS$DCLCMH                   ********  GX    04
STORE_STEP                       00001364 R      04        SYS$DELMBX                   ********  GX    04
STP0                             0000003D R      04        SYS$DLCEFC                   ********  GX    04
STP1                             000000AF R      04        SYS$EXIT                     ********  GX    04
STP10                            000002DC R      04        SYS$FAO                      ********   X    04
STP11                            00000312 R      04        SYS$FAOL                     ********  GX    04
STP12                            0000035C R      04        SYS$GETCHN                   ********  GX    04
STP13                            000003FE R      04        SYS$GETDEV                   ********  GX    04
STP14                            0000044C R      04        SYS$GETDVI                   ********  GX    04
STP15                            000004B3 R      04        SYS$GETMSG                   ********  GX    04
STP16                            000004E2 R      04        SYS$HIBER                    ********  GX    04
STP17                            0000056C R      04        SYS$PUTMSG                   ********  GX    04
STP18                            000005DC R      04        SYS$QIO                      ********  GX    04
STP19                            00000679 R      04        SYS$QIOW                     ********  GX    04
STP2                             000000D0 R      04        SYS$SETAST                   ********  GX    04
STP20                            000006F9 R      04        SYS$SETPRN                   ********  GX    04
STP21                            00000734 R      04        SYS$SETPRV                   ********  GX    04
STP22                            00000778 R      04        SYS$TRNLOG                   ********  GX    04
STP23                            000007B0 R      04        SYS$WAITFR                   ********  GX    04
STP24                            00000816 R      04        SYS$WAKE                     ********  GX    04
STP25                            00000860 R      04        SYSTEST_DIR                  000004AF R      03
STP26                            000008CF R      04        TEST_DATA                    00000250 R      02
STP27                            000008F2 R      04        TEST_MOD_BEGIN               00000019 R      02
STP28                            00000913 R      04        TEST_MOD_FAIL                0000002A R      02
STP29                            00000981 R      04        TEST_MOD_NAME                00000000 R      02
```

```
TEST_MOD_NAME_D                  00000009 R      02
TEST_MOD_SUCC                    0000001F R      02
TMD_ADDR                         0000004C R      03
TMN_ADDR                         00000048 R      03
TOPSYS                           000004D2 R      03
TOPSYS_DIR                       000004E4 R      03
TOTAL_CHAN                       00001338 R      04
TPID                             00000000 R      03
UETP$_DATAER                   = 00748010
UETP$_SATSMS                   = 007480D9
UETP$_TEXT                     = 00741133
UM                               000001FE R      02
WARNING                        = 00000000

                                              +-----------------+
                                              ! Psect synopsis !
                                              +-----------------+

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (     0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD  NOWRT NOVEC BYTE
$ABS$                           00000000  (     0.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
RODATA                          000002EC  (   748.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD  NOWRT NOVEC LONG
RWDATA                          000004FB  (  1275.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD    WRT NOVEC LONG
SATSSS01                        00001D0D  (  7437.)  04 (  4.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD    WRT NOVEC LONG

                                            +------------------------+
                                            ! Performance indicators !
                                            +------------------------+

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                     33    00:00:00.09    00:00:00.48
Command processing                112    00:00:00.63    00:00:01.54
Pass 1                           1286    00:00:34.12    00:01:00.91
Symbol table sort                   0    00:00:03.70    00:00:04.43
Pass 2                            846    00:00:08.61    00:00:10.90
Symbol table output                18    00:00:00.28    00:00:00.78
Psect synopsis output               3    00:00:00.03    00:00:00.04
Cross-reference output              0    00:00:00.00    00:00:00.00
Assembler run totals             2301    00:00:47.47    00:01:19.09

The working set limit was 1800 pages.
210140 bytes (411 pages) of virtual memory were used to buffer the intermediate code.
There were 120 pages of symbol table space allocated to hold 2297 non-local and 50 local symbols.
2069 source lines were read in Pass 1, producing 48 object records in Pass 2.
105 pages of virtual memory were used to define 100 macros.
```

```
                                        +--------------------------+
                                        ! Macro library statistics !
                                        +--------------------------+

Macro library name                      Macros defined
------------------                      --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2                   82
_$255$DUA28:[SHRLIB]UETP.MLB;1                      13
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                       2
_$255$DUA28:[SYSLIB]STARLET.MLB;2                    0
TOTALS (all libraries)                              97

2752 GETS were required to define 97 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SATSSS01/OBJ=OBJ$:SATSSS01 MSRC$:SATSSS01/UPDATE=(ENH$:SATSSS01)+EXECML$/LIB+SHRLIB$:UETP/LIB
```

0421 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SATSSS08
LIS

SATSSS05
LIS

SATSSS22
LIS

SATSSS07
LIS

SATSSS01
LIS